Adaptive High-Order Discretization of the Reynolds-Averaged Navier-Stokes(RANS) Equations

By

Cheng Zhou

Submitted to the Department of Aerospace Engineering and the
Graduate Faculty of the University of Kansas
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Committee members

Dr. Z.J. Wang, Chairperson

Dr. Saeed Farokhi

Dr. Ray Taghavi

Dr. Xuemin Tu

Dr. Zhongquan Charlie Zheng

Date defended: August 29, 2016

The Dissertation Committee for Cheng Zhou certifies
that this is the approved version of the following dissertation :

Adaptive High-Order Discretization of the Reynolds-Averaged Navier-Stokes(RANS) Equations

Dr. Z.J. Wang, Chairperson

Date approved:

Abstract

The use of high-order methods to compute turbulent flows governed by the Reynolds-averaged Navier-Stokes (RANS) equations is an active research topic in the computational fluid dynamics (CFD) community. However, it is well known that high-order methods for the non-smooth turbulence modeling equations are difficult to converge to the steady-state because of the numerical stiffness. The objective of this work is to develop a robust and efficient high-order discretization that can simulate turbulent flows governed by the Reynolds-Averaged Navier-Stokes equations, which involves the development of high-order space discretization of robust turbulence modeling equations, the improvement of time integration strategy, and the application of effective mesh adaptation methods.

In the present study, correction procedure via reconstruction (CPR) high-order discretization is developed to solve the Reynolds-averaged Navier-Stokes (RANS) equations with the modified Spalart and Allmaras (SA) model. In this model, the non-dimensional length scale depends on the distance to the nearest wall. To compute the distance of each solution point in the domain to the nearest curved polynomial wall boundaries, the CPR high-order discretization is extended to solve the Eikonal equation. On the other hand, to improve time integration strategy for the simulation of turbulent flows, the present work carried out a comparative study of several implicit time integration schemes to determine which is the most efficient, robust and general scheme. Additionally, an adjoint-based adaptive mesh refinement method is utilized to minimize the output error. Numerical results show that, to achieve a certain level of accuracy, the adaptive CPR discretization of the RANS equations with the SA model

saves orders of magnitude in terms of number of degrees of freedom comparing to the numerical results of uniform mesh refinement, when applied to the simulations of turbulent flows.

Acknowledgements

First and foremost, I would like to show my deep appreciation and gratitude to my advisor Dr. Z.J. Wang. His guidance during the past five years lays the foundation of this thesis work. He is always there to keep me on track when I have different ideas and to motivate me when I was going to give up. The most important thing I learned from him is to rise to challenges: whenever a challenge comes, never say "I can't", say "Let me try". It is my honer being part of his research group.

I would like to thank my committee members, Dr. Xuemin Tu, Dr. Charlie Zheng, Dr. Saeed Farokhi, and Dr. Ray Taghavi. Their questions and comments during our meetings are intelligent and inspiring. I thank Professor Xuemin Tu for the detailed introduction in our discussions about the parallel preconditioners and implicit time integration schemes. I thank Professor Charlie Zheng for the inspiring questions during our discussions about the turbulence models. I thank Professor Saeed Farokhi and Professor Ray Taghavi for the challenging questions during my comprehensive oral exam and for agreeing to be my committee members.

Special thanks to Dr. Lei Shi for offering me generous help on C++ techniques and challenging questions during our discussions about the difficulties in research. My friend Yanan Li deserves recognition for listening to my frustrations and giving me interesting perspectives on work and life. Thanks to my group members for being awesome officemates.

Without the endless support from my husband Ruxin and our parents, and the happiness brought by my daughter Felicia, I cannot make my dream come true.

Contents

List of Figures

List of Tables

Nomenclature

α	Angle of attack
$\alpha_{j,f,l}$	Lifting constants independent of the solution variables
χ	Nondimensional turbulence working variable
δ_i	Correction field for element i
ε	Coefficient of artificial viscosity in Eikonal equation
γ	Ratio of specific heats
κ_{SA}	Scaling factor for the Spalart-Allmaras equation
$\mathbf{F}$	Flux vector
$\mathbf{F}^n$	Normal flux vector
$\mathbf{F}^n_{com}$	Common flux at the interface
$\mathbf{Q}$	State vector
$\mathbf{R}$	Residual of the linear system
$\mathbf{S}$	Source term
$\mathscr{J}$	Engineering output
μ	Fluid dynamic viscosity

μ_t Turbulence eddy viscosity

ν Kinematic viscosity

Ω Computational domain

ω^n Under-relaxation parameter for time step n

Π Projection operator

Ψ Modified nondimensional turbulence working variable in the Spalart-Allmaras model

ρ Density

τ_{ij} Fluid viscous stress tensor for Newtonian fluid

$\tilde{\nu}$ Turbulence working variable in Spalart-Allmaras model

$\tilde{S}$ Production term of the Spalart-Allmaras model

$\vec{\omega}$ Vorticity vector

$\vec{v}$ Front propagation velocity

C_d Drag coefficient

C_f Skin friction coefficient

C_l Lift coefficient

C_p Surface pressure coefficient

d Distance to the nearest wall

E Specific total energy per unit mass

L Lagrange interpolation polynomial

M_0 Mach number at freestream

P	Pressure
p	Polynomial order of numerical discretization
P^k	Space of polynomials of degree k
Pr	Prantle number
Pr_t	Turbulence Prantle number
Re	Reynolds number
S_f	Face area of face f
T	Temperature
u	Velocity in the x-direction
v	Velocity in the y-direction
V_i	Non-overlapping element
W	Weighting function
w	Velocity in the z-direction
y^+	Nondimensional distance to the nearest wall
J	Jacobian matrix

Chapter 1

Introduction

1.1 Motivation

Computational Fluid Dynamics (CFD) has undergone tremendous development as a discipline in the past few decades and is used routinely to complement the wind tunnel experiments in the design of aircraft. Nearly all the production codes used in the aerospace community are first or second order accurate finite volume, finite difference, or finite element methods. They are capable of running on small clusters with overnight turnaround time to achieve engineering accuracy (e,g., 5% error) for Reynolds-averaged Navier-Stokes (RANS) simulations.

Whereas second-order methods have been the workhorse for CFD in industry, there are still many flow problems that are too expensive or out of their reach. For example, the flow over a helicopter[78]. In order to obtain an engineering accuracy level prediction of the aerodynamic forces on the helicopter body, the tip vortices generated by the rotor need to be resolved for a long distance before hitting the body, because of their strong influence on the aerodynamic loading on the helicopter body. Since the numerical dissipation and dispersion of the first and second order methods are relatively large to resolve the unsteady vortices, the globally mesh refinement is usually used to solve these problems. Furthermore, aerodynamic flows can exhibit singularities that introduce additional challenges for output prediction, the mesh resolution requirement for the

flow makes such simulations too expensive even on modern supercomputers. Thus, for vortex dominated flows, high-order methods are needed because of their superior accuracy and efficiency. Ultimately, the mesh and order adaptation methods (hp-adaptation), which allow the flow field to dictate the local order of accuracy and grid resolution, will be the most efficient approach.

1.2 Background and Objective

High-order CFD methods have received considerable attention in the research community because of the potential in delivering higher accuracy with lower cost than low order methods. The use of high-order methods to compute turbulent flows governed by the Reynolds-averaged Navier-Stokes (RANS) equations is an active research topic in the computational fluid dynamics(CFD) community. It is well known that high-order methods for the non-smooth turbulence modeling equations are difficult to converge to the steady-state because of the numerical stiffness [17, 54, 55, 57]. The objective of this work is to develop a robust and efficient high-order discretization that can simulate turbulent flows with the Reynolds-Averaged Navier-Stokes equations. This objective involves the development of high-order space discretization of robust turbulence modeling equations, the improvement of time integration strategy, and the application of effective mesh adaptation methods.

1.2.1 Reynolds-Averaged Navier-Stokes Equations with Spalart-Allmaras Model

The one equation Spalart-Allmaras(SA) model [68] is widely used in the application of high-order methods to the RANS equations, because of its simple implementation and the only one turbulence working variable. But in the simulation of turbulent flows, the turbulence working variable always decreases abruptly at the edge between the turbulent and laminar regions. This non-smooth behavior often leads to negative values of the turbulence working variable, which subsequently causes solver failure. To alleviate the stability issues caused by the negative turbulence

working variable, several researchers [55–57] have proposed a variety of modifications for the SA turbulence model. The modified SA turbulence model investigated in [16, 54, 56] has been demonstrated to significantly improve the robustness of high-order simulations, which is employed by the present work.

Another challenge in the application of high-order methods to the RANS equations is the computation of wall distance. It is well known that even for finite volume methods, the computation of nearest wall distance is an expensive task for a complex geometry[27] . Several problems arising in high-order schemes significantly increase this computational complexity, such as the large number of solution points in higher-order elements, the curved, piecewise polynomial wall boundaries[10] , and the higher-order accuracy of wall distance required by its smoothness in higher-order methods[27] .

In the present study, a high-order accurate solver has being developed for the Eikonal equation to compute the nearest distance to the wall. In the design of this solver, we adopt a front propagation velocity, which is the normalization of the distance gradient. After adding the virtual time step and changing the convective term into a conservative form, the high-order CPR framework can be directly used to solve the equation. We also add a diffusive term to stabilize the equation for geometric singularities. The diffusive coefficient is proportional to the distance at each solution point, so that on one hand the geometry singularity can be resolved during the simulation and on the other hand the diffusive term vanishes at the wall boundary, which provides accurate near wall distance for turbulence models. This approach is efficient because the searching for the nearest wall is avoided, especially for 3 dimensional problems. Also, it is straightforward to parallelize this solver on CPU clusters.

1.2.2 Time Integration Schemes

Improving computational efficiency is of great importance to the CFD community, and the use of implicit time integration schemes is one solution since they can advance the solution with significantly larger time steps comparing with the explicit schemes. On the other hand, due to high-order

space discretizations, very large Jacobian matrices are produced, which make implicit schemes expensive in both memory and CPU time. Upon this trade-off between memory and efficiency, many implicit schemes have been developed, either to decrease memory requirement[36] or to accelerate convergence to steady state [6, 7, 36, 60]. However, it is unclear which is the most efficient, robust and general. The present work performs a comparative study of several well-known or recently developed implicit time-integration schemes with the Correction Procedure via Reconstruction (CPR) high-order discretization for unstructured grids. The following implicit schemes are considered in the study, namely the Lower-Upper Symmetric Gauss-Seidel (LU-SGS), a matrix-free GMRES solver with LU-SGS as a preconditioner, and a GMRES linear equation solver with different preconditioners available in PETSc (Portable, Extensible Toolkit for Scientific Computation) library[6] , such as Jacobian, ILU and ILU(1) and line preconditioner.

The original LU-SGS approach was developed by Yoon and Jameson[84], which employs a special first-order approximation in linearization of the left-hand side to reduce the block diagonal matrices to diagonal matrices. Through forward and backward sweeps in one Gauss–Seidel iteration, all of the off-diagonal matrices still contribute to the implicit operator. As a result, LU-SGS does not require any extra memory compared to explicit schemes, but can drive a steady solution to convergence much faster than explicit schemes. It was also found that the convergence rate could degrade considerably for viscous flow problems, especially after several orders of convergence[22]. To further improve the convergence rate, Chen and Wang[22] and Jameson and Caughey[44] developed a block (preconditioned) non-linear LUSGS (BLU-SGS) approach. For a wide variety of flow problems, BLU-SGS demonstrated low storage requirement (only the diagonal block matrix is stored), and faster convergence to steady state than the original LUSGS. Details on a very compact form of the non-linear BLU-SGS used in this study can be found in Reference[70].

In the conventional GMRES approach, the full implicit matrix needs to be stored including the lower and upper matrices, which may require prohibitive storage in large 3D computations. This deficiency can be alleviated by various matrix-free implementations of GMRES[60]. Every successful GMRES iterative approach needs a good preconditioner. To achieve fast convergence,

the preconditioner should be "close" to the implicit operator, but much easier to invert. Bassi and Rebay successfully developed a preconditioned GMRES approach[63] for the DG method[10] to solve compressible flow. A matrix-free Krylov approach was developed by Rasetarinera and Hussaini[60] with an efficient LU-SGS preconditioner. This implementation demonstrates a considerable saving in storage compared to the standard GMRES.

In the present work, according to a series of numerical experiments on the BLU-SGS approach and GMRES with BLU-SGS as the preconditioner (GMRES-BLUSGS), it is found that GMRES-BLUSGS, which only requires storing main block diagonal matrix, is a good compromise between efficiency and storage requirement. However, it failed to achieve convergence with the CPR high-order discretization for a flat plate boundary layer flow. To remedy this problem, we resort to the preconditioned GMRES linear solvers available in the PETSc library (PGMRES), which generally requires the derivative of the Jacobian matrix that can be approximated with first order finite differences or analytically computed with dual number[59]. PETSc provides a robust and flexible suit of data-structure-neutral numerical routines for Newton-like methods, which enable various storage schemes and solvers through a single user interface[6, 7, 39].

To improve the robustness of high-order discretization, the present work also implemented the matrix reordering algorithm for the block-ILU preconditioner, proposed by Diosady et al.[25]. This reordering algorithm based on the lines of maximum coupling between elements, is demonstrated to be superior to standard reordering techniques such as nested dissection, one-way dissection, quotient minimum degree, reverse Cuthill-Mckee. In addition,in terms of solution update, Ceze et.[21] showed that line-searches optimization can improve the global convergence property when solving nonlinear systems, which is also applied in the present study.

1.2.3 Adjoint-based Error Estimation and Mesh Adaptation

Even though the CFD community has developed various numerical methods for the simulation of turbulent flows governed by the Reynolds-Averaged Navier-Stokes equations, including finite difference, finite volume and finite element methods, it was found that different codes on

different grid families can exhibit different convergence characteristics for a relatively simple turbulent flow over the NACA0012 airfoil[15, 26, 53]. It appears that the local grid resolution around geometric singularities, such as the sharp trailing edge, plays a more critical role than discretization schemes[26]. Given that those finest grids in Ref. [26] are not sufficient to establish an asymptotic convergence order, a solution based hp-adaptation method offers the best promise [28, 38, 66, 72, 76].

In this thesis, a dual-consistent high-order correction procedure via reconstruction method (CPR) is utilized to discretize the RANS equations with the modified one-equation SA model, and an adjoint-based adaptive method is applied to dynamically distribute the computing power to under-resolved regions for predicting an engineering output such as lift or drag with minimum output error.

The effectiveness of adaptive methods highly depends on the accuracy of the error estimation. The adaptation criteria for error estimation are classified into four major types of adaptation criteria: gradient or feature based[8, 14, 37, 81], residual-based[3, 5, 18, 32, 45, 67], entropy variable-based[30, 31], and adjoint-based[13, 20, 29, 33, 34, 38, 49, 58, 71, 72, 75, 83]. The feature-based criterion such as large gradient cannot provide an universal and robust error estimation[71, 85]. The residual-based error indicator is defined locally on each element and has achieved some successes; however, it can lead to false refinements in convection-dominated flow. In the entropy variable-based method, entropy variables can be interpreted as the dual solution for the output of entropy balance in the whole domain. It can be obtained directly from the state variables without solving extra adjoint equations. The dual-weighted residual method proposed by Becker and Rannacher [12] relates a specific functional output directly to the local residual by solving an additional adjoint equation. It can capture the error propagation effects inherent in the hyperbolic equations. This kind of adjoint-based error indicator has been shown very effective in driving a hp-adaptation procedure to obtain an accurate prediction of the functional outputs [13, 20, 33, 34, 76]. In the present method, a dual-consistent high-order CPR is utilized to compute the discrete adjoint solution associated to the engineering outputs, e.g., the lift or drag coefficient, and derive the

output-based local error indicator.

The mesh refinement procedure is driven by the automated output-based adaptation. The candidate elements for adaptation can be modified by enriching its solution order or subdividing its element or resizing its grid. There are three generally classified ways to increase the discretization resolution: h-refinement, r-refinement and p-refinement. For h-refinement, subdivision is performed locally for each candidate element to increase the total degrees of freedom. R-refinement or the moving mesh method keeps the total number of nodes the same but moves the location of the grid locally or globally [40]. With p-refinement, the local degree of approximation polynomial is modified. In the present study, we only perform hierarchical element refinements from a very coarse mesh.

1.3 Thesis Overview

This thesis addresses the development of an adaptive high-order accurate discretization of the RANS equations. The specific contributions are as follows:

- Derived a discrete form of the RANS equations with the SA model for the CPR framework.
- Derived a discrete form of the Eikonal equation for the CPR framework, and verified the accuracy on several geometries with analytic solution.
- Evaluated different implicit time integration schemes and preconditioners. Determined the most efficient implicit solver, preconditioner and solution update strategy for high-order simulation of turbulent flows.
- Implemented parallel implicit time integration schemes for distributed memory system and optimized the performance.
- Implemented the 2D discrete RANS equations with the SA model, and discrete Eikonal equation for the CPR method. Performed the accuracy and efficiency study with other flow solvers and experimental data.

- Extended the CPR discretization of the RANS equations with the SA model, and the Eikonal equation to 3D parallel and adaptive approach.

- Incorporated the adjoint-based error estimation and mesh adaptation method for the parallel simulation of turbulent flows using the CPR method.

- Demonstrated the importance of the high-order CPR method upon simulations of turbulent flows and applied it to a wide range of engineering applications.

Chapter 2 describes the Correction Procedure via Reconstruction method used for the spatial discretization, and the application to the RANS equations with the Spalart-Allmaras model and the Eikonal equation. The comparison of time-integration schemes and the optimization of the solution update are presented in Chapter 3. Chapter 4 describes the adjoint-based error estimation and the mesh adaptation methods, followed by the numerical verification and validation of the proposed CPR discretization of the RANS equations. Finally, Chapter 5 present conclusions and suggestions for future work.

Chapter 2

Space Discretization

2.1 The High-Order CPR Method

For the sake of completeness, the CPR formulation is briefly reviewed. The CPR method was originally developed by Huynh[41, 42] , and extended to simplex and hybrid elements by Wang and Gao[79] . Other recent developments are reviewed in Ref. [43]. The CPR formulation has some remarkable properties: it is easy to understand, efficient to implement and can recover several well known methods such as the discontinuous Galerkin (DG),the spectral volume method (SV) and the spectral difference methods (SD). The degrees-of-freedoms (DOFs) are the conservative variables at a predefined nodal set named solution points(SPs), where the differential form of the governing equations is solved. In the present study, the solution points are chosen as the Gauss quadrature points. Details about the CPR approach can be found in Ref. [77]. Here we derive from a hyperbolic conservation law which can be written as

$$\frac{\partial \mathbf{Q}}{\partial t} + \nabla \cdot \mathbf{F}(Q) = 0 \tag{2.1}$$

with proper initial and boundary conditions, where $\mathbf{Q}$ is the state vector, and $\mathbf{F}(\mathbf{Q})$ is the flux vector. Assume that the computational domain Ω is discretized into N non-overlapping elements $\{V_i\}_{i=1}^{N}$. Let W be an arbitrary weighting function or test function. The weighted residual formulation of

Eq. (2.1) on element V_i can be expressed as

$$\int_{V_i} (\frac{\partial \mathbf{Q}}{\partial t} + \nabla \cdot \mathbf{F}(Q)) W \, \mathrm{d}V = 0. \tag{2.2}$$

Apply integration by parts to the flux divergence, we can get

$$\int_{V_i} \frac{\partial \mathbf{Q}}{\partial t} W \, \mathrm{d}V + \int_{\partial V_i} W \mathbf{F}(\mathbf{Q}) \cdot \vec{n} \, \mathrm{d}S - \int_{V_i} \nabla W \cdot \mathbf{F}(\mathbf{Q}) \, \mathrm{d}V = 0. \tag{2.3}$$

Let $\mathbf{Q}_i$ be an approximate solution to the analytic solution $\mathbf{Q}$ on V_i, which belongs to the space of polynomials of degree k or less, i.e.,$\delta_i \in P^k(V_i)$. The numerical solution $\mathbf{Q}_i$, for the moment, is required to satisfy Eq. (2.3) as

$$\int_{V_i} \frac{\partial \mathbf{Q}_i}{\partial t} W \, \mathrm{d}V + \int_{\partial V_i} W \mathbf{F}(\mathbf{Q}_i) \cdot \vec{n} \, \mathrm{d}S - \int_{V_i} \nabla W \cdot \mathbf{F}(\mathbf{Q}_i) \, \mathrm{d}V = 0. \tag{2.4}$$

Obviously the surface integral is not properly defined because the numerical solution is discontinuous across element interfaces. Following the idea used in the Godunov method [35, 48], the normal flux term in Eq. (2.4) is replaced with a common Riemann flux, e.g., in Ref. [50, 61, 62]

$$\mathbf{F}^n(\mathbf{Q}_i) \equiv \mathbf{F}(\mathbf{Q}_i) \cdot \vec{n} \approx \mathbf{F}^n_{com}(\mathbf{Q}_i, \mathbf{Q}_{i+}, \vec{n}), \tag{2.5}$$

where $\mathbf{Q}_{i+}$ denotes the solution outside the current element V_i. Instead of Eq. (2.4), the approximate solution is required to satisfy

$$\int_{V_i} \frac{\partial \mathbf{Q}_i}{\partial t} W \, \mathrm{d}V + \int_{\partial V_i} W \mathbf{F}^n_{com} \, \mathrm{d}S - \int_{V_i} \nabla W \cdot \mathbf{F}(\mathbf{Q}_i) \, \mathrm{d}V = 0. \tag{2.6}$$

Applying integration by parts again to the last term of the above LHS, we obtain

$$\int_{V_i} \frac{\partial \mathbf{Q}_i}{\partial t} W \, \mathrm{d}V + \int_{V_i} W \nabla \cdot \mathbf{F}(\mathbf{Q}_i) \, \mathrm{d}V + \int_{\partial V_i} W [\mathbf{F}^n_{com} - \mathbf{F}^n(\mathbf{Q}_i)] \, \mathrm{d}S = 0. \tag{2.7}$$

The test space here has the same dimension as the solution space, and is chosen in a manner to guarantee the existence and uniqueness of the numerical solution. To obtain a differential formulation, the elimination of the test function is critical. The boundary integral above is cast as a volume integral via the introduction of a "correction field" on V_i, $\delta_i \in P^k(V_i)$,

$$\int_{V_i} W \delta_i \, dV = \int_{\partial V_i} W [\mathbf{F}^n] \, dS, \tag{2.8}$$

where $[\mathbf{F}^n] = \mathbf{F}^n_{com} - \mathbf{F}^n(Q_i)$ is the normal flux difference. Substituting Eq. (2.8) into Eq. (2.7), we have

$$\int_{V_i} (\frac{\partial \mathbf{Q}_i}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{Q}_i) + \delta_i) W \, dV = 0. \tag{2.9}$$

If the flux vector is a linear function of the state variable, then $\nabla \cdot \mathbf{F}(\mathbf{Q}_i) \in P^k$. In this case, the terms inside the square bracket are all elements of P^k . Because the test space is selected to ensure a unique solution, Eq. (2.9) is equivalent to

$$\frac{\partial \mathbf{Q}_i}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{Q}_i) + \delta_i = 0. \tag{2.10}$$

For nonlinear conservation laws, $\nabla \cdot \mathbf{F}(\mathbf{Q}_i)$ is usually not an element of P^k. In this case, the most obviously choice is to project $\nabla \cdot \mathbf{F}(\mathbf{Q}_i)$ into P^k. Denote $\Pi(\nabla \cdot \mathbf{F}(\mathbf{Q}_i))$ as a projection of $\nabla \cdot \mathbf{F}(Q_i)$ to P^k. One choice is

$$\int_{V_i} \Pi(\nabla \cdot \mathbf{F}(\mathbf{Q}_i)) W \, dV = \int_{V_i} \nabla \cdot \mathbf{F}(\mathbf{Q}_i) W \, dV. \tag{2.11}$$

Then Eq. (2.9) reduces to

$$\frac{\partial \mathbf{Q}_i}{\partial t} + \Pi(\nabla \cdot \mathbf{F}(\mathbf{Q}_i)) + \delta_i = 0. \tag{2.12}$$

Note that for δ_i defined by Eq. (2.8), if $W \in P^k$, Eq. (2.12) is equivalent to the discontinuous-Galerkin (DG) formulation, at least for linear conservation laws; if W belongs to another space, the resulting δ_i is different. We obtain a formulation corresponding to a different method such as the spetral volume (SV) method.

The differential equation is solved at solution points (SPs) $\{\vec{r}_{i,j}\}$ (j varies from 1 to K). Then Eq. (2.12) holds true at the SPs, i.e.,

$$\frac{\partial \mathbf{Q}_{i,j}}{\partial t} + \Pi_j(\nabla \cdot \mathbf{F}(\mathbf{Q}_i)) + \delta_{i,j} = 0, \tag{2.13}$$

The efficiency of the CPR approach hinges on how the correction field δ_i and the projection $\Pi(\nabla \cdot \mathbf{F}(\mathbf{Q}_i))$ are computed. Two efficient approaches on how to compute $\prod_j(\nabla \cdot \mathbf{F}(\mathbf{Q}_i))$ are developed in Ref. [79], namely, Lagrange polynomial approach(LP) and chain rule approach(CR). The computation of $\prod_j^v(\nabla \cdot \mathbf{F}^v(\mathbf{Q}_i, \mathbf{R}_i))$ in the present study follows the LP approach. To compute δ_i, we define k+1 points named flux points (FPs) along each interface, where the normal flux differences are computed. Then we approximate (for nonlinear conservation laws) the normal flux difference $[\mathbf{F}^n]$ with a degree k interpolation polynomial along each interface

$$[\mathbf{F}^n]_f \approx \mathbf{I}_k[\mathbf{F}^n]_f \equiv \sum_l [\mathbf{F}^n]_{f,l} L_l^{FP}, \tag{2.14}$$

where f is a face (or edge in 2D) index, and l is the FP index, and L_l^{FP} is the Lagrange interpolation polynomial based on the FPs in a local interface coordinate. Substituting Eq. (2.14) to the linearized equation Eq. (2.8), the correction at the SPs can be written as

$$\delta_{i,j} = \frac{1}{|V_i|} \sum_{f \in \partial V_i} \sum_l \alpha_{j,f,l} [\mathbf{F}^n]_{f,l} S_f, \tag{2.15}$$

where $\alpha_{j,f,l}$ are lifting constants independent of the solution variables, S_f is the face area, $|V_i|$ is the volume of V_i. Note that the correction for each solution point, namely $\delta_{i,j}$, is a linear combination of all the normal flux differences on all the faces of the cell. Conversely, a normal flux difference at a flux point on a face, say (f,l) results in a correction at all solution points j of an amount $\alpha_{j,f,l}[\mathbf{F}^n]_{f,l} S_f / |V_i|$.

Next we apply the CPR formulation to the compressible Reynolds Averaged Navier-Stokes

(RANS) equations, which can be written as

$$\frac{\partial \mathbf{Q}(t)}{\partial t} + \nabla \cdot (\mathbf{F}(\mathbf{Q}) - \mathbf{F}_{\mathbf{v}}(\mathbf{Q}, \nabla \mathbf{Q})) = \mathbf{S}(\mathbf{Q}, \nabla \mathbf{Q}) \tag{2.16}$$

with additionally the viscous flux vector F_v and the source term vector $\mathbf{S}$.

First, following Ref. [9] , we introduce a new variable $\mathbf{R} = \nabla \mathbf{Q}$. Let $\mathbf{R}_i$ be an approximation of $\mathbf{R}$ on element V_i. The CPR discretization of Eq. (2.19) can be expressed as,

$$\frac{\partial \mathbf{Q}_{i,j}}{\partial t} + \prod_j (\nabla \cdot \mathbf{F}(\mathbf{Q}_i)) - \prod_j^v (\nabla \cdot \mathbf{F}^v(\mathbf{Q}_i, \mathbf{R}_i)) + \frac{1}{|J_{i,j}|} \sum_{f \in \partial V_i} \sum_l \alpha_{j,f,l} ([\mathbf{F}^n]_{f,l} - [\mathbf{F}^{v,n}]_{f,l}) S_{f,l} = \mathbf{S}(\mathbf{Q}_{i,j}, \mathbf{R}_{i,j}),$$

$$\mathbf{R}_{i,j} = \nabla \mathbf{Q}_{i,j} + \frac{1}{|J_{i,j}|} \sum_{f \in \partial V_i} \sum_l \alpha_{j,f,l} [\mathbf{Q}^{com} - \mathbf{Q}_i]_{f,l} \vec{n}_{f,l} S_{f,l}, \tag{2.17}$$

where $|J_{i,j}|$ is the determinant of the Jacobian matrix and,

$$\begin{aligned} [\mathbf{F}^n] &= \mathbf{F}^n_{com} - \mathbf{F}(\mathbf{Q}_i), \\ [\mathbf{F}^{v,n}]_f &= \mathbf{F}^v(\mathbf{Q}^{com}_f, \nabla \mathbf{Q}^{com}_f) \cdot \vec{n}_f - \mathbf{F}^v(\mathbf{Q}_i, \mathbf{R}_i)|_f \cdot \vec{n}_f, \end{aligned} \tag{2.18}$$

with $\mathbf{Q}^{com}_f$ and $\nabla \mathbf{Q}^{com}_f$ the common solution and gradient on interface f respectively, and $\mathbf{Q}_i$ is the solution within cell V_i on the flux point (FP) l of face f. $\mathbf{F}^n$ is the normal flux on the interface. $\mathbf{F}^n_{com}$ denotes a common flux on the interface reconstructed by any Riemann Solver. In the current study, we use the Roe Riemann flux to compute the inviscid common flux $\mathbf{F}^n_{com}$. Various schemes for viscous fluxes differ in how the common solution $\mathbf{Q}^{com}_f$ and the common gradient $\nabla \mathbf{Q}^{com}_f$ are defined. In the present study, we employ the BR2 [11] scheme to compute the common solution and gradient on interfaces. More other schemes can be found in [23, 69, 73, 80].

2.2 Discretization of the RANS Equations with the SA Model

2.2.1 The Governing Equations

The conservation form of the compressible Reynolds Averaged Navier-Stokes (RANS) equations with the modified one-equation Spalart-Allmaras turbulence model [16, 54, 56] can be written as

$$\frac{\partial \mathbf{Q}(t)}{\partial t} + \nabla \cdot (\mathbf{F}(\mathbf{Q}) - \mathbf{F_v}(\mathbf{Q}, \nabla \mathbf{Q})) = \mathbf{S}(\mathbf{Q}, \nabla \mathbf{Q}) \tag{2.19}$$

with the proper boundary and initial conditions within a domain Ω. Here, the vector of conservative variables $\mathbf{Q}$, the convective flux vector $\mathbf{F}$,the viscous flux vector F_v and the source term vector $\mathbf{S}$ are given by:

$$\mathbf{Q} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \\ \rho \tilde{\nu} \end{pmatrix}, \mathbf{F^x} = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ u(E+P) \\ \rho u \tilde{\nu} \end{pmatrix}, \mathbf{F^y} = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + P \\ v(E+P) \\ \rho v \tilde{\nu} \end{pmatrix},$$

$$\mathbf{F_v^x} = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} + c_p(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t})\frac{\partial T}{\partial x} \\ \frac{1}{\sigma}(\mu + \mu\Psi)\frac{\partial \tilde{\nu}}{\partial x} \end{pmatrix}, \mathbf{F_v^y} = \begin{pmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ u\tau_{yx} + v\tau_{yy} + c_p(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t})\frac{\partial T}{\partial y} \\ \frac{1}{\sigma}(\mu + \mu\Psi)\frac{\partial \tilde{\nu}}{\partial y} \end{pmatrix}, \tag{2.20}$$

$$\mathbf{S} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ c_{b1}\tilde{S}\rho\nu\Psi + \frac{1}{\sigma}[c_{b2}\rho\nabla\tilde{\nu}\cdot\nabla\tilde{\nu}] - c_{w1}\rho f_w(\frac{\nu\Psi}{d})^2 - \frac{1}{\sigma}\nu(1+\Psi)\nabla\rho\cdot\nabla\tilde{\nu} \end{pmatrix}$$

where the ρ, P, E are respectively the density, pressure and specific total energy per unit mass, u,v denote the Cartesian velocity. ν denotes the kinematic viscosity and $\tilde{\nu}$ represents the turbulence working variable defined in the modified SA model. Then the pressure P is given by,

$$P = (\gamma - 1)\left(\rho E - \frac{1}{2}\rho(u^2 + v^2)\right) \tag{2.21}$$

where $\gamma = 1.4$ is the ratio of specific heats. Define velocity vector $\mathbf{u} = (u, v)$, then the fluid viscous stress tensor for Newtonian fluid τ is defined as,

$$\tau_{ij} = (\mu + \mu_t)\left(\frac{\partial \mathbf{u}_i}{\partial \mathbf{x}_j} + \frac{\partial \mathbf{u}_j}{\partial \mathbf{x}_i} - \frac{2}{3}\frac{\partial \mathbf{u}_k}{\partial \mathbf{x}_k}\delta_{ij}\right) \tag{2.22}$$

where δ_{ij} is the Kronecker delta. μ refers to the fluid dynamic viscosity, and μ_t refers to the turbulence eddy viscosity defined by the SA model as,

$$\begin{aligned} \mu_t &= \begin{cases} \rho\tilde{\nu} f_{v1} & \text{if } \tilde{\nu} \geq 0 \\ 0 & \text{if } \tilde{\nu} < 0 \end{cases} \\ f_{v1} &= \frac{\Psi^3}{\Psi^3 + C_{v1}^3} \end{aligned} \tag{2.23}$$

For the source term $\mathbf{S}$ of Eq. (2.20), the production term of the modified SA model $\tilde{S}$ is given as[56] ,

$$\begin{aligned} \tilde{S} &= \begin{cases} S + \hat{S} & \text{if } \hat{S} \geq -c_{v2}S \\ S + \frac{S(c_{v2}^2 S + c_{v3}\hat{S})}{(c_{v3} - 2c_{v2})S - \hat{S}} & \text{if } \hat{S} < -c_{v2}S \end{cases} \\ S &= \sqrt{\vec{\omega}\cdot\vec{\omega}} \\ \hat{S} &= \frac{\nu\Psi}{\kappa_t^2 d^2} f_{v2} \\ f_{v2} &= 1 - \frac{\Psi}{1 + \Psi f_{v1}} \end{aligned} \tag{2.24}$$

where $\vec{\omega} = \nabla \times \mathbf{u}$ is the vorticity vector. The destruction term coefficients are given by,

$$\begin{aligned} r &= min\left[\frac{\nu\Psi}{\tilde{S}\kappa_t^2 d^2}, 10\right] \\ g &= r + c_{w2}(r^6 - r) \\ f_w &= g\left(\frac{1+c_{w3}^6}{g^6+C_{w3}^6}\right)^{1/6} \end{aligned} \tag{2.25}$$

where d denotes the distance to the nearest wall at a specific location. The parameter Ψ is designed for high-order discretization schemes to remove the effects of negative turbulence working variable on the robustness of the turbulence model. This parameter is given as,

$$\begin{aligned} \Psi &= \begin{cases} 0.05\ln(1+e^{20\chi}) & \text{if } \chi \leq 10 \\ \chi & \text{if } \chi > 10 \end{cases} \\ \chi &= \frac{\tilde{\nu}}{\nu} \end{aligned} \tag{2.26}$$

when $\tilde{\nu}$ goes negative, the parameter Ψ can prevent instabilities by turning off the production, destruction and dissipation terms. Finally, the constants in the modified SA model are given as,

$$\begin{aligned} &c_{b1} = 0.1335,\ c_{b2} = 0.622,\ \sigma = 2/3,\ \kappa_t = 0.41,\ P_r = 0.72,\ P_rt = 0.9, \\ &c_{w1} = c_{b1}/\kappa^2 + (1+c_{b2})/\sigma,\ c_{w2} = 0.3,\ c_{w3} = 2,\ c_{v1} = 7.1,\ c_{v2} = 0.7,\ c_{v3} = 0.9 \end{aligned} \tag{2.27}$$

2.2.2 Dynamic Scaling of the Discrete Equation

The discretization form as Eq. (2.17) can result in ill-conditioning implicit systems, especially for practical cases with Reynolds number of $10^6 \sim 10^7$. In this regime, $\tilde{\nu}/\nu_\infty$ typically ranges from 10^4 to 10^5. In order to alleviate the ill-conditioning and improve the floating point precision, Ceze[21] introduced a constant scaling factor for $\tilde{\nu}$, which is non-dimensionalized $\rho\tilde{\nu}$ by a factor lager than the physical viscosity. For further improvement, we propose a dynamic scaling of the conservative variable $\rho\tilde{\nu}$ so that its numeric value is of the same order of magnitude as other

conservative variables. The dynamic scaling is expressed as,

$$(\rho\tilde{\nu})' = \frac{\rho\tilde{\nu}}{\kappa_{SA}\mu_\infty}, \tag{2.28}$$

where $(\rho\tilde{\nu})'$ is the scaled conservative variable that is computed by the solver, μ_∞ is the farfield dynamic viscosity and κ_{SA} is a scaling factor which is dynamically updated according to the solution of each time step as follows,

$$\kappa_{SA} = \begin{cases} 10^{-4}, & \text{if } \kappa_{SA} < 10^{-4} \\ \frac{\kappa_{SA}(\rho\tilde{\nu})'_{max}}{(\rho u)_{max}}, & \text{if } 10^{-4} \leq \kappa_{SA} \leq 10^{4} \\ 10^{4}. & \text{if } \kappa_{SA} > 10^{4}, \end{cases} \tag{2.29}$$

and initially, $\kappa_{SA} = 1$.

In order to exemplify the dynamic scaling, Fig. 2.1 displays the residual history for a subsonic turbulent flow over NACA0012 airfoil at $Re_c = 6 \times 10^6, M_0 = 0.15, \alpha = 10^\circ$, with a $p = 1$ space discretization. As expected, the dynamic scaling factor κ_{SA} improves the convergence rate.

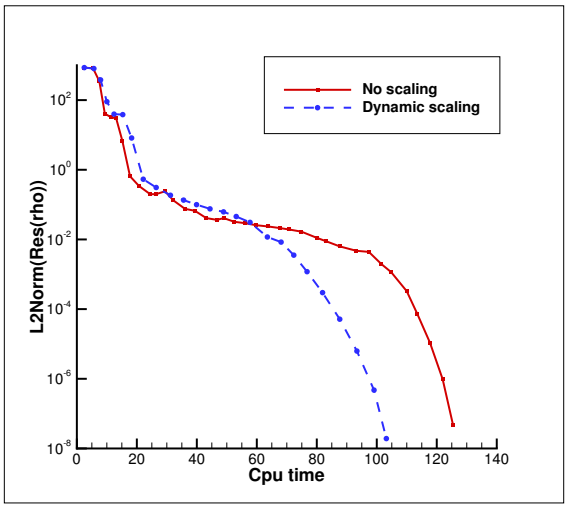


Figure 2.1: Comparison of residual convergence with and without dynamic scaling.

2.2.3 Boundary Conditions

The wall boundary is a no-slip adiabatic wall, which means $\nabla T \cdot \mathbf{n} = 0$ and $\mathbf{Q}_b = (\mathbf{Q}_1, 0, 0, \mathbf{Q}_4, 0)$. To prevent the occurrence of an apparent transition reported by Crivellini et al[24] , the freestream turbulent viscosity is set to $\tilde{\nu}_\infty = 3\nu$, instead of the ideal condition $\tilde{\nu}_\infty = 0$. This value is considered perfectly acceptable by Spalart and Allmaras [4, 68].

2.3 Eikonal Equation for Wall Distance Computation

2.3.1 Background

A challenge in the application of high-order methods to RANS equations is the computation of the nearest distance to walls. A search based algorithm is difficult to parallelize on massively parallel clusters. In the present study, we obtain the wall distance by solving the Eikonal equation

$$\|\nabla d\| = 1, \tag{2.30}$$

with homogeneous Dirichlet boundary conditions, $d = 0$, imposed on the wall boundary. In order to obtain high-order accuracy for the nearest wall distance, Liu et al.[51] proposed a finite element discretization of Eq. (2.30) for internal flow problems. Schoenawa[64] extended the discretization in Ref. [51] to external flow geometries by adding a stabilization based on the streamline diffusion and artificial viscosity. However, this finite element discretization is different from the widely used discontinuous high-order discretizations, eg. Discontinuous Galerkin (DG). Thus most of the stabilizing procedure does not transfer to discontinuous high-order methods.

In the following, we develop a new high-order solver for the Eikonal equation by adopting a front propagation velocity[82] to convert Eq. (2.30) into the conservative form, and directly use the high-order CPR method to solve the equation. For more complicated flow geometries, the CPR discretization requires a further stabilization with a diffusive term.

2.3.2 Discretization

First we square Eq. (2.30) and define a front propagation velocity $\vec{v} = \nabla d / \|\nabla d\|$ to obtain

$$\vec{v} \cdot \nabla d = 1. \tag{2.31}$$

Adding the virtual time step and converting the convective term into the conservative form lead to

$$\frac{\partial d}{\partial t} + \nabla \cdot (\vec{v} d) - d \nabla \cdot \vec{v} = 1. \tag{2.32}$$

To alleviate the non-linearity of Eq. (2.32), we freeze the front propagation velocity $\vec{v}$ for 200 time steps before the first update of velocity field, and then update the velocity field every 50 time steps. Within those frozen steps, an artificial viscosity $\mu = \varepsilon d$ is introduced to further stabilize the simulation,

$$\frac{\partial d}{\partial t} + \nabla \cdot (\vec{v} d) - \mu \nabla \cdot \nabla d = 1. \tag{2.33}$$

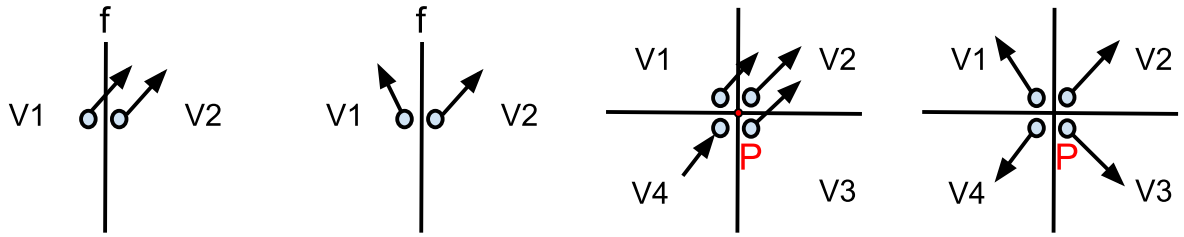
Let $Q = d$, $\mathbf{F} = \vec{v} d$, $\mathbf{F}_v = \nabla d$, and $S = 1$, Eq. (2.33) is similar to the conservative form of RANS-SA equations as Eq. (2.19),

$$\frac{\partial Q(t)}{\partial t} + \nabla \cdot (\mathbf{F}(Q) - \mu \mathbf{F}_\mathbf{v}(Q, \nabla Q)) = S, \tag{2.34}$$

with the same CPR discretization in Eq. (2.17), the Eikonal equation can be efficiently solved with high-order accuracy.

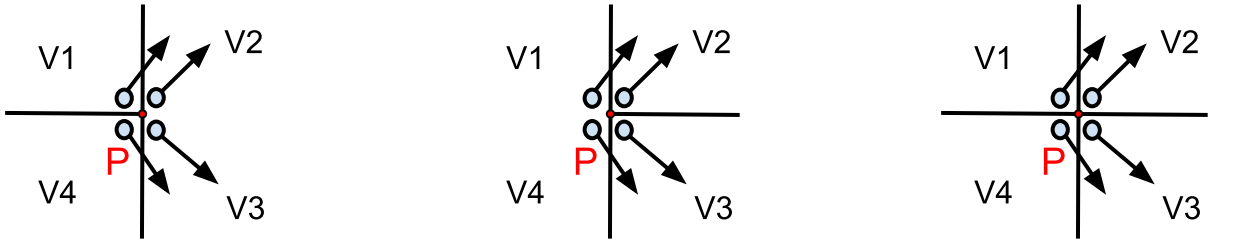
The Gauss-Lobatto points are used as solution points(SPs) and flux points(FPs) in the discretization. The numerical experiments indicated that the front propagation velocity has to be unique on interfaces and vertices. In the present work, the velocity of SPs and FPs on wall boundaries always use the geometric normal on the boundary. On other interfaces, due to the front propagation nature of the Eikonal equation, we choose the upwind velocity as the unique front propagation velocity. Otherwise the averaged velocity is used if there is no upwind velocities, see Fig. 2.2(a)(b). On the vertices, the weighted average of all interface velocities connected to the

vertex forms the unique front propagation velocity on that vertex, see Fig. 2.2(c)(d), so that the parallelization does not affect the uniqueness, as Fig. 2.3 illustrates.



(a) $\vec{v}_f = \vec{v}_1$ (b) $\vec{v}_f = (\vec{v}_1 + \vec{v}_2)/2$ (c) $\vec{v}_p = \frac{1}{4}(\vec{v}_1 + \vec{v}_3 + \vec{v}_4 + \vec{v}_4)$ (d) $v_p = \frac{1}{4}(\frac{\vec{v}_1+\vec{v}_2}{2} + \frac{\vec{v}_2+\vec{v}_3}{2} + \frac{\vec{v}_3+\vec{v}_4}{2} + \frac{\vec{v}_1+\vec{v}_4}{2})$

Figure 2.2: Computation of the unique front propagation velocity on interfaces and vertices

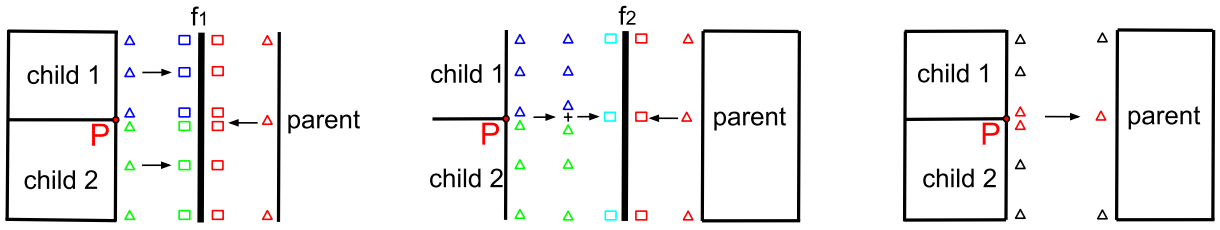


(a) $w_p|_l = 0.5+0.5+1.=2.$ $\vec{v}_p|_l = \frac{1}{2}(0.5\vec{v}_1 + 0.5\vec{v}_4 + \frac{\vec{v}_1+\vec{v}_4}{2})$

(b) $w_p|_r = 0.5+1.+0.5=2.$ $\vec{v}_p|_r = \frac{1}{2}(0.5\vec{v}_1 + \frac{\vec{v}_2+\vec{v}_3}{2} + 0.5\vec{v}_4)$

(c) $\vec{v}_p = \frac{1}{4}(\vec{v}_1 + \frac{\vec{v}_2+\vec{v}_3}{2} + \vec{v}_4 + \frac{\vec{v}_1+\vec{v}_4}{2}) = \frac{\vec{v}_p|_l \cdot w_p|_l + \vec{v}_p|_r \cdot w_p|_r}{w_p|_l + w_p|_r}$

Figure 2.3: Computation of the unique front propagation velocity in parallelization



(a) Child side: compute unique velocity $\vec{v}_f$ on f_1

(b) Parent side: compute unique velocity $\vec{v}_f$ on f_2

(c) Interpolation from child side to parent side, after the weighted average of velocity on the hanging node

Figure 2.4: Computation of the unique front propagation velocity on non-conforming faces

Algorithm 1 Weighted average velocity

for each wall boundary face f_{wall} **do**
    unique velocity $\vec{v}_{f_{wall}} \leftarrow \vec{n}_{f_{wall}}$ ▷ use the geometric normal as velocity
end for
for all vertex p **do**
    $\vec{v}_p \leftarrow 0.$ ▷ initialize velocity for all vertices
end for
for all other interface f **do**
    compute unique velocity $\vec{v}_f$ ▷ see Fig. 2.2(a)(b) and Fig. 2.4(a)(b)
    if $f \in \{domain\ interfaces, non-conforming\ faces\}$ **then** ▷ set weight for each face
        $w_f \leftarrow 0.5$
    else
        $w_f \leftarrow 1.$
    end if
    for end points($p_1,\ p_2$) of face f **do** ▷ summarize velocity and weight on vertices, Fig. 2.2(c)(d)
        if p is not on the wall boundary **OR** $f \in \{f_{wall}\}$ **then**
            $\vec{v}_p \leftarrow \vec{v}_p + \vec{v}_f$
            $w_p \leftarrow w_p + w_f$
        end if
    end for
end for
for all vertex p **do**
    $\vec{v}_p \leftarrow \vec{v}_p / w_p$ ▷ weighted averagy velocity on vertices
end for
Update $\vec{v}_f$ of non-conforming faces ▷ see Fig. 2.4(c)
Normalize $\vec{v}_f$ and $\vec{v}_p$, and assign the unique velocity to collocating SPs and FPs

Non-conforming faces require a special treatment to determine the unique front propagation velocity. The faces of child cells and the face of parent cell are treated as separated faces, and different mortar faces are created when computing a unique velocity on those mortar faces. For a child side, the velocity polynomial of the parent side is interpolated to mortar face f_1. Then the unique velocity computed on f_1 is used as the front propagation velocity of child side faces. For a parent side, the velocity polynomials of both children are averaged on the hanging node p before the projection to mortar face f_2. Then the unique velocity computed on f_2 is used as the front propagation velocity of the parent side face. Fig. 2.4 describes this treatment. After the weighted average of velocities on all vertices, including hanging nodes, the unique velocity on the child side is interpolated to the parent side, so that the two sides still keep the same velocity polynomial.

Algorithm 1 summarizes how to determine the unique front propagation velocity.

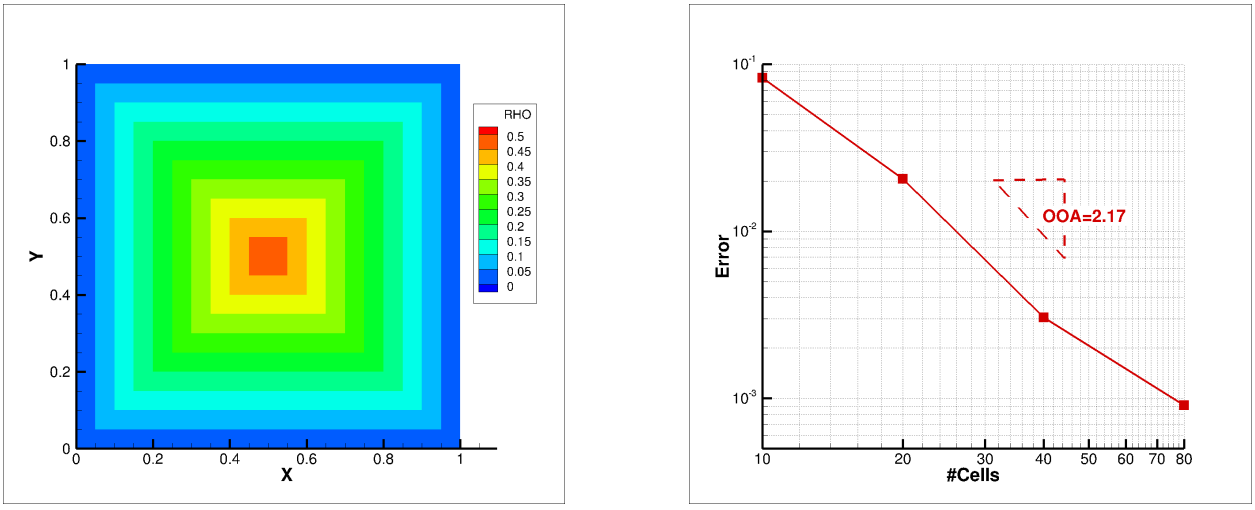
2.3.3 Numerical Verification for CPR Eikonal Solver

In order to demonstrate the CPR Eikonal solver, an internal and three external flow geometries are tested in the following. The initial condition is the distance to a single point in the field, like the center of gravity of a geometry or the moment reference point of an airfoil.

The first test is the square domain $[0,1]^2 \subset \mathbf{R}^2$ surrounded by wall boundaries. This internal flow geometries has the analytic solution given by,

$$d(x,y) \;=\; min(x,1-x,y,1-y). \tag{2.35}$$

4 meshes were tested, with $5\times 5, 10\times 10, 20\times 20, 40\times 40$ elements, respectively. And numerical experiments suggested $\varepsilon = 0.001, 0.001, 0.01, 0.01$, respectively. We chose $\{0,0\} \subset \mathbf{R}^2$ as the reference point, and solved the discretization with $p = 1$ only. Figure 2.5 shows the solution on 40×40 mesh, and the errors on different meshes.



(a) the p=1 wall distance solution

(b) the errors on different mesh

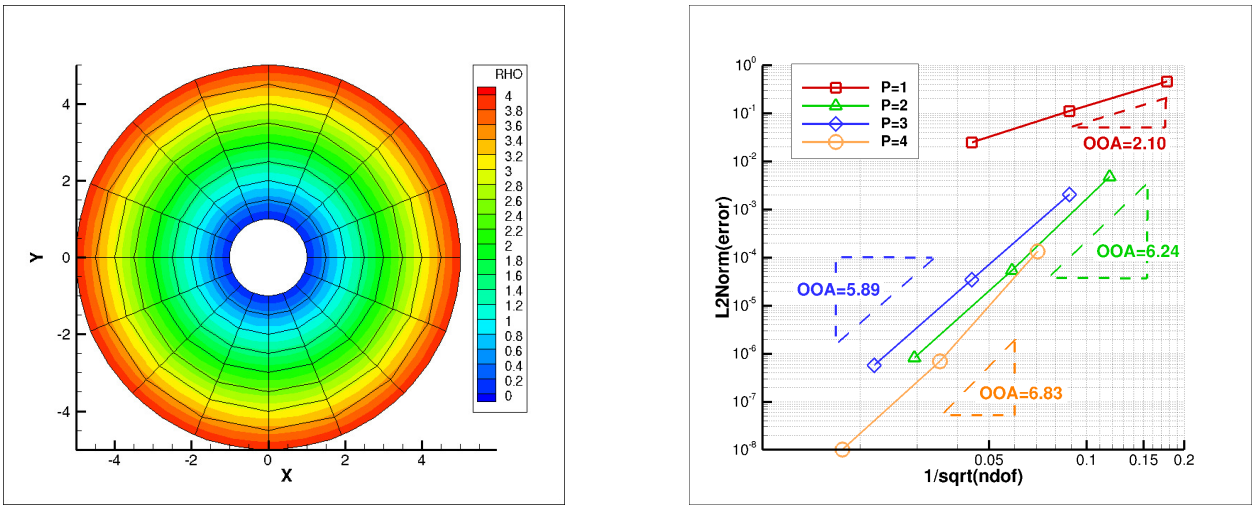
Figure 2.5: Wall distance in square domain on a mesh with 1600 elements

The second case is a ring domain $r \in [1,5]$ with inner cylinder as wall boundary and outer cylinder as extrapolation boundary. We chose this case to verify the CPR solver for the Eikonal

equation by computing the order of accuracy (OOA) based on the analytic solution given by,

$$d(x,y) \;=\; \sqrt{x^2+y^2}-1. \tag{2.36}$$

Three meshes were tested with 8, 32, 128 fourth-order curved elements, respectively. For this simple case, $\varepsilon = 0$. $\{0,0\} \subset \mathbf{R}^2$ was chosen as reference point. Figure 2.6 shows the solution and errors on different meshes with different discretization orders. We solved the discretization with $p = 1,2,3,4$ and achieved super convergence for this smooth geometry and smooth solution problem.

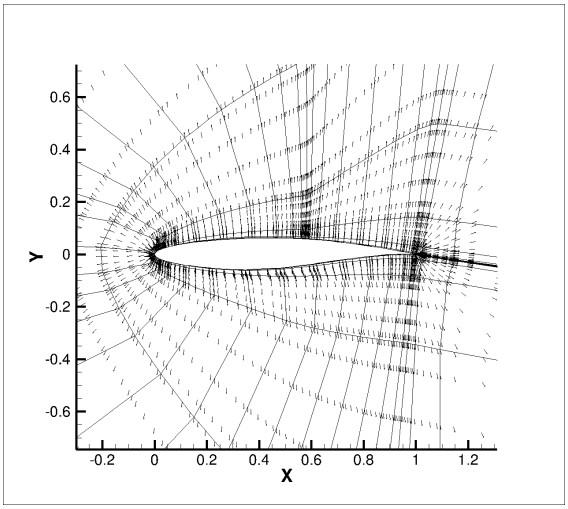


(a) the p=4 wall distance solution

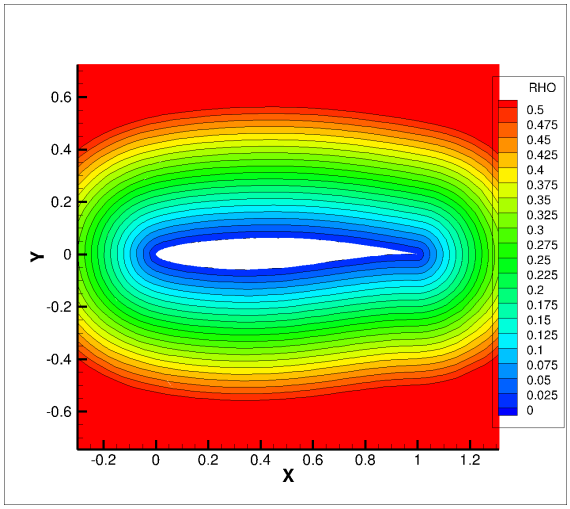
(b) the errors on different meshes and different orders

Figure 2.6: Wall distance on the mesh with 128 $p = 4$ curved elements

Next we considered a mesh around the RAE2282 airfoil with 506 curved elements of degree 4. The mesh is from the 1^{st} International Workshop on High-Order CFD Methods[1] , generated by Deconinck. We chose the quarter chord point $(0.25,0) \subset \mathbf{R}^2$ as the reference point. On this unsymmetrical mesh, we solved the discretization with $p = 1,2...5$. Numerical experiments suggested $\varepsilon = 0.,0.,0.01,0.01,0.01$ respectively. For computational efficiency, each converged lower order solution is used as initial condition for the next higher-order solution. All the residuals drop 8 orders of magnitude. For this geometry, there is no analytic wall distance available. Figure 2.7(b) shows the wall distance solution with $p = 5$ near the airfoil.



(a) the connective velocity on solution point



(b) the p=5 wall distance solution

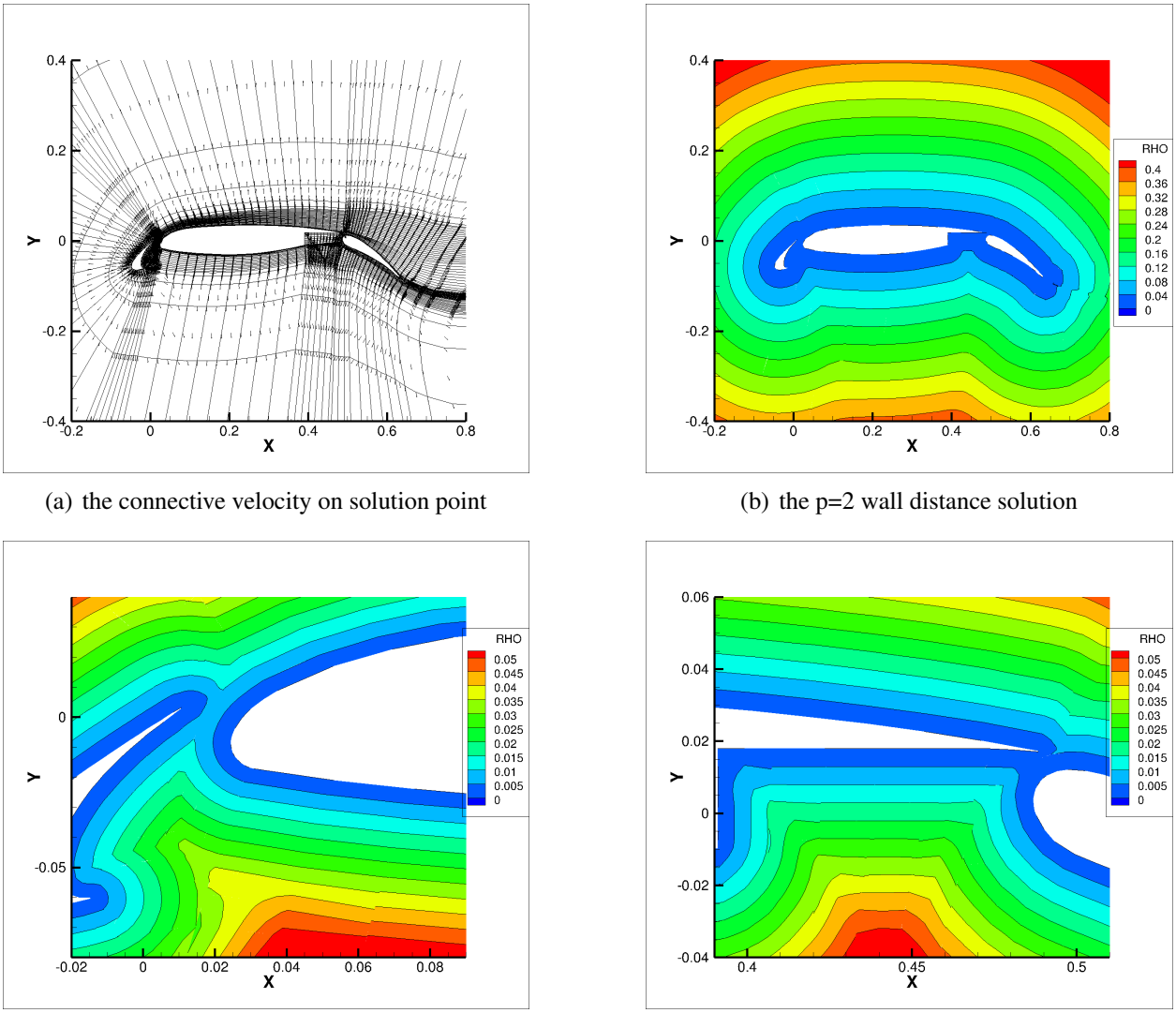
Figure 2.7: Wall distance around the RAE2282 airfoil on the mesh with 506 curved elements

As a final example, we tested the CPR Eikonal solver on a mesh around the high-lift multi-element 30P30N airfoil, with 4070 curved elements of degree 4. The mesh is also from the Workshop[1] , generated by Ceze. Again, we chose $(0.25,0) \subset \mathbf{R}^2$ as the reference point. On this mesh, we solve the discretization with $p = 1,2,3$. Each converged lower order solution is used as initial condition for the next higher-order solution. Numerical experiments show $\varepsilon = 0.$ is acceptable for $p = 1$ so that the residual drops 16 orders of magnitude. For higher-order discretizations, the minimum value of $\varepsilon = 0.01$ drops the residual by 10 orders of magnitude. Figure 2.8 shows the wall distance solution with $p = 2$ around the airfoil.

2.4 Test Cases

In this section, we apply the CPR discretization of the RANS equations with the SA model to two steady flow problems.

The initial conditions of all the $p = 1$ solutions have been set to uniform freestream. For computational efficiency, each converged lower order solution is used as the initial flow field of the next higher order solution. All the results in the present work are obtained with the standard GMRES linear solver available in the PETSc library[6]. The Jacobian matrix J is computed analytically



(a) the connective velocity on solution point

(b) the p=2 wall distance solution

(c) the wall distance solution near the slat

(d) the wall distance solution near the flap

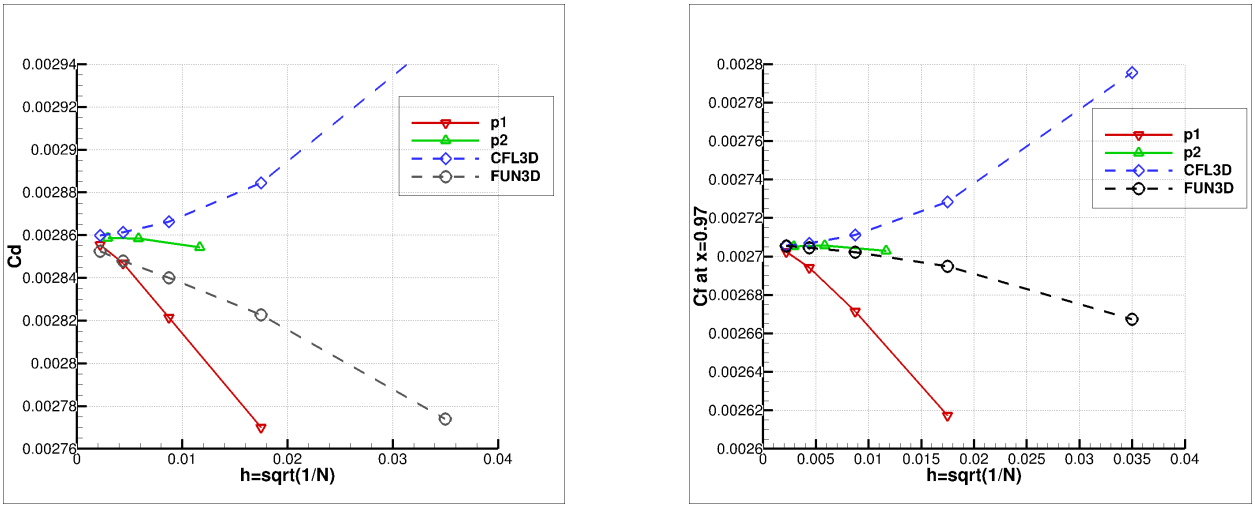
Figure 2.8: Wall distance around the 30P30N airfoil on the mesh with 4070 curved elements

using the dual number[59] , which has been already verified. According to authors' experience on laminar flow[86], the preconditioner is chosen as the incomplete lower-upper factorization, ILU(1), in the PETSc library. The parameters of the restarted GMRES solver is set to 90 Krylov space vectors, 200 maximum iterations and 10^{-8} relative convergence tolerance. The initial time step is $CFL = 1$. The CFL grows following the line search method reported in[21] . For computation efficiency, the CFL is increased to infinity (1.e10 in present study) after the residual drops 6 orders of magnitude.

2.4.1 Flat Plate, $M_0 = 0.2$,$Re_c = 5 \times 10^6$

First we consider a turbulent flow over a flat plate at Mach number $M_0 = 0.2$ with Reynolds number $Re = 5 \times 10^6$ based on the length of 1. The main aim of this case is to verify the accuracy of the CPR discretization by comparing with other results on the Turbulence Modeling Resource (TMR) webpage[2].

Four meshes with 34×24, 68×48, 136×96, 272×192 elements, respectively, are solved with $p = 1, 2$ discretization. The coarsest mesh gives an approximate average $y^+ \approx 1.7$ over the plate. Figure 2.9 shows the convergence history of drag coefficient and skin friction coefficient at $x = 0.97$, compared with CFL3D and FUN3D[47] results. The converged results agree with CFL3D and FUN3D within 0.1 count. The convergence history of the skin friction over the flat plate is depicted in Figure 2.10, which demonstrates that when the mesh is finer and discretization order is higher, the result gets closer and closer to the CFL3D result. For discretization with $p = 2$, the solutions of the finest two meshes have excellent agreement with each other. Figure 2.11 compares the extracted non-dimensional eddy viscosity at $x = 0.97$. The $p = 1$ result shows some oscillation at the leading edge, while the $p = 2$ results shows excellent agreement with the CFL3D and FUN3D results.



(a) the drag coefficient C_d

(b) the skin friction coefficient C_f at x=0.97

Figure 2.9: Turbulent flow over a flat plate with $M_0 = 0.2, Re_c = 5 \times 10^6$

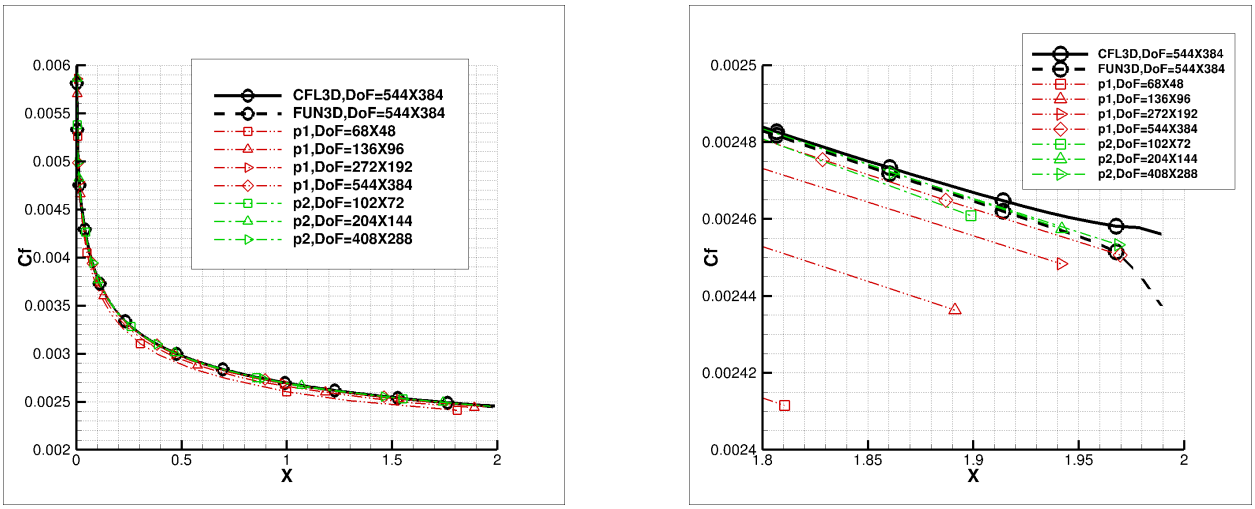


Figure 2.10: The skin friction coefficient of turbulent flow over a flat plate at $M_0 = 0.2, Re_c = 5 \times 10^6$

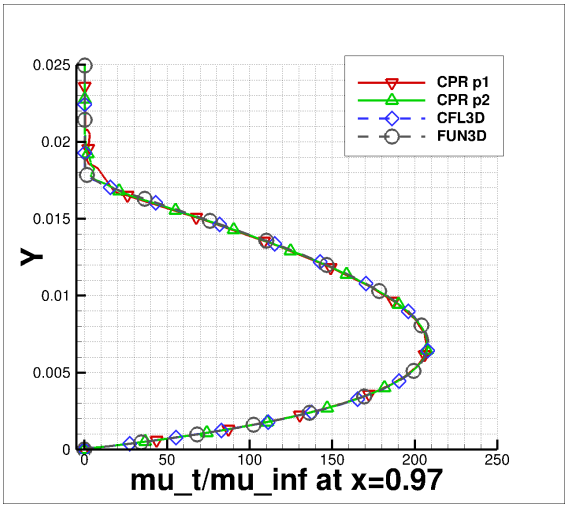


Figure 2.11: The non-dimensional eddy viscosity of turbulent flow over a flat plate at $M_0 = 0.2, Re_c = 5 \times 10^6, x = 0.97$ with 136×96 elements

In order to numerically verify the accuracy of the present discretization, Richardson Extrapolation[19] is performed for the drag coefficient and skin friction coefficient at the position $x = 0.97$. Fig. 2.12(a) displays the computed apparent order of the CPR second-order and third-order discretizations along with the NASA's CFL3D and FUN3D codes. The CPR discretization didn't achieve full order of accuracy because of the singularity at the leading point of the flat plate. The approximate relative fine-grid error presented in Fig. 2.12(b) indicates that third-order CPR scheme gives much less error than the lower order schemes, which demonstrated the necessity of high-order discretizations. Fig. 2.12(c)(d) show the extrapolated relative fine-grid error and fine-grid convergence index, the green bars show almost 0 value, which means the mesh and order independent results have been achieved. Fig. 2.13 describes the similar conclusion for the skin friction coefficients at the position of $x = 0.97$

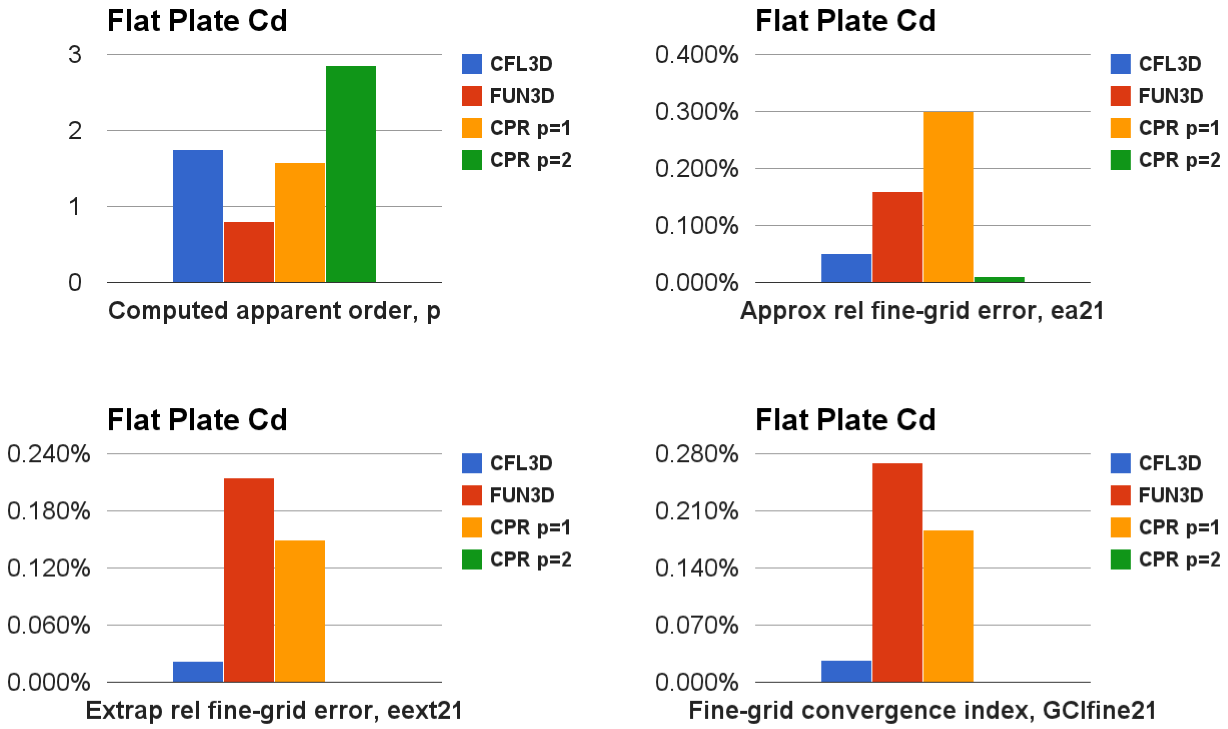


Figure 2.12: Richardson Extrapolation of the drag coefficient C_d

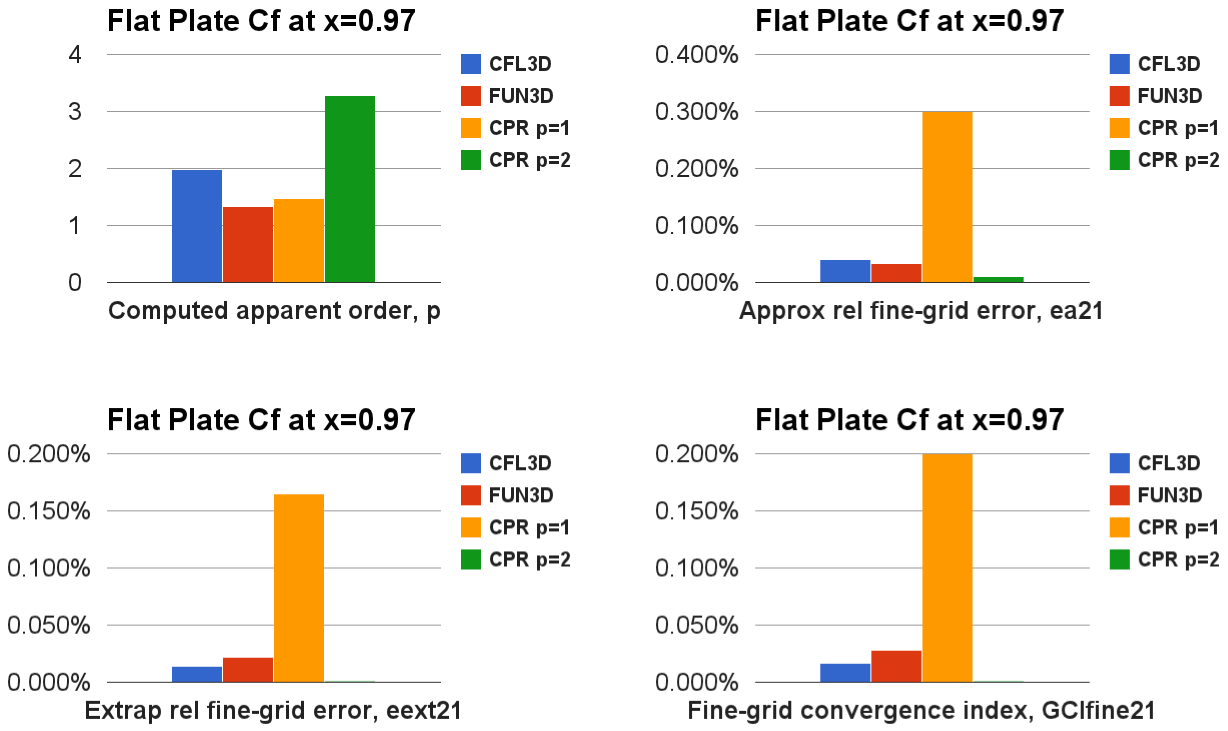


Figure 2.13: Richardson Extrapolation of the skin friction coefficient C_f

2.4.2 NACA 0012, $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 0^\circ, 10^\circ, 15^\circ$

The next test case is a turbulent flow over the NACA0012 airfoil at Mach number $M_0 = 0.15$, Reynolds number $Re = 6 \times 10^6$ based on cord length, with angles of attack $\alpha = 0^\circ, 10^\circ, 15^\circ$. This case is used as a validation case of CFD codes on the TMR webpage[2] , by comparing all the CFD results with the experimental results. Note that, an airfoil at $\alpha = 15^\circ$ experiences unsteady flow, mainly due to unsteady separation point. Thus the validation is based on time-averaged forces, e.g., lift and drag.

The farfield boundary is located almost 500 chords away from the airfoil. We test three C-type meshes $h = 0, 1, 2$ with $54 \times 23, 108 \times 46, 216 \times 92$ fourth-order curved elements respectively. The first layer grid of the coarsest mesh gives $y^+ \approx 10$. Figure 2.14 shows two views of the 54×23 mesh. Discretization orders of $p = 1, 2, 3, 4$ are computed for this case. The residuals for all cases drop 10 orders of magnitude. The computational results are compared with experimental data as well as the CFL3D[47] results. The CFL3D results used here are obtained with 897×257 elements

and is considered as benchmark solutions for this case.

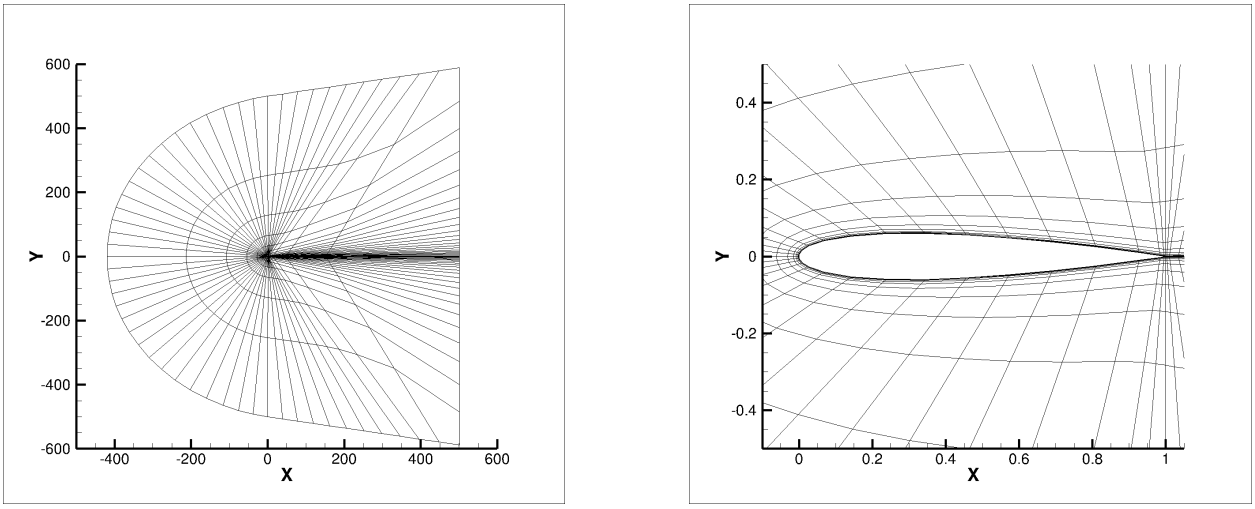


Figure 2.14: The 54×23 mesh of NACA0012 airfoil

Figure 4.7 shows the convergence of the drag and lift coefficients for different angles of attack with increased discretization orders for medium mesh $h = 1$. The $p = 1$ results are always far away from the time-averaged experimental data. With higher orders of accuracy, both the C_d and C_l appear to converge to the time-averaged experimental data and CFL3D results.

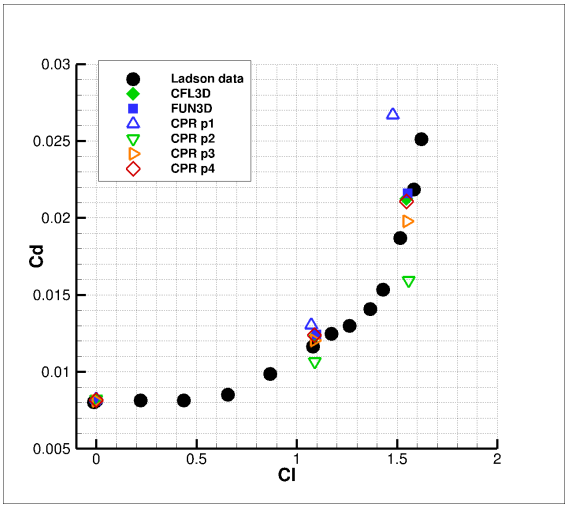


Figure 2.15: $C_l vs. C_d$ for turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 0^\circ, 10^\circ, 15^\circ$ on medium mesh $h = 1$

Figures 2.16-2.18 depict the computational results for $\alpha = 0^\circ$. Figure 2.16 demonstrates that the mesh and order independent results are obtained for the drag coefficient. Figure 2.17 and

Figure 2.18 compared the computed surface pressure coefficient and skin friction coefficient with the experimental data and CFL3D results. The present simulation shows excellent agreement in surface pressure coefficient with experimental and CFL3D results as expected. It appears more difficult to compute the skin friction accurately. On the coarse mesh $h = 0$, even the highest discretization order $p = 4$ cannot accurately capture the distribution of the skin friction coefficient. But for medium mesh $h = 1$, as the discretization order becomes higher, the results show improved agreement with CFL3D. In addition, there is an excellent agreement between the $p = 3$ and $p = 4$ CPR schemes and the CFL3D results. On the finest mesh $h = 2$, the $p = 2$ CPR scheme demonstrates better agreement with the CFL3D results than those on the coarse meshes.

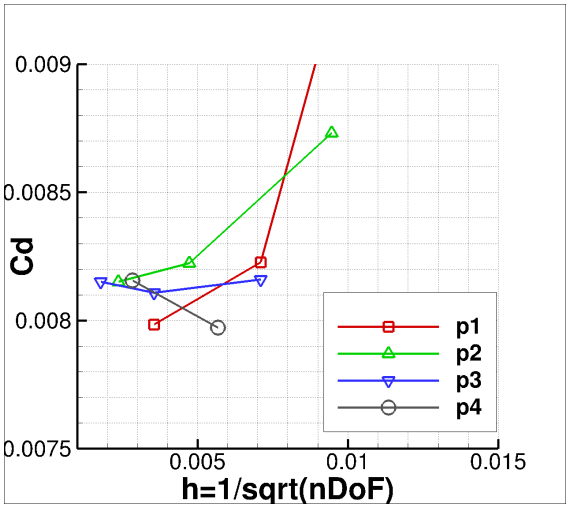
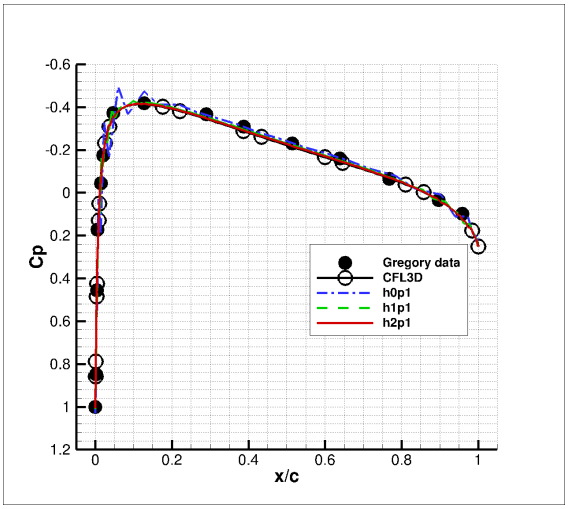
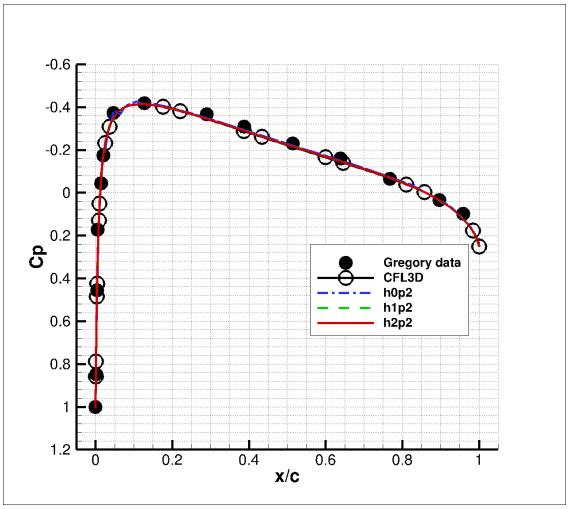


Figure 2.16: Drag coefficient for turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 0^\circ$

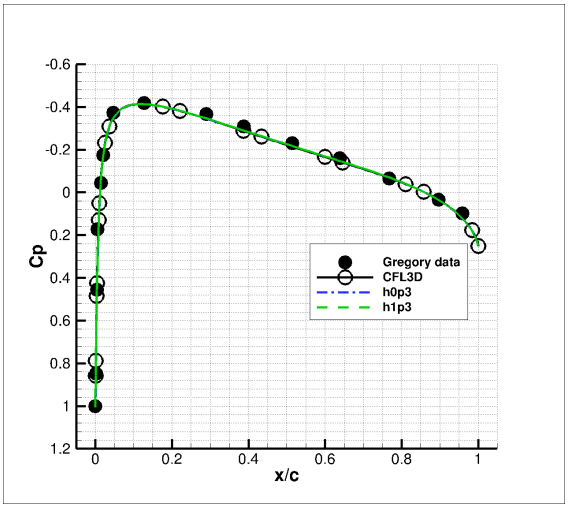
Two similar studies are performed for $\alpha = 10^\circ$ and $\alpha = 15^\circ$, as displayed in Figures 2.19-2.20 and Figures 2.21-2.22, respectively. Similarly, the $p = 1$ CPR scheme is not accurate enough to capture the distribution of the skin friction coefficient, even on the finest mesh. On the coarse mesh $h = 0$, the higher-order scheme shows better agreement with the CFL3D result. And the $p = 3$ and $p = 4$ schemes give excellent agreement with the CFL3D results on the medium mesh. However, the lift and drag coefficients presented in Fig. 2.19 and Fig. 2.21 indicate that, we are still far away from the mesh and order independent results in terms of the engineering outputs even if a relative fine mesh is already used.



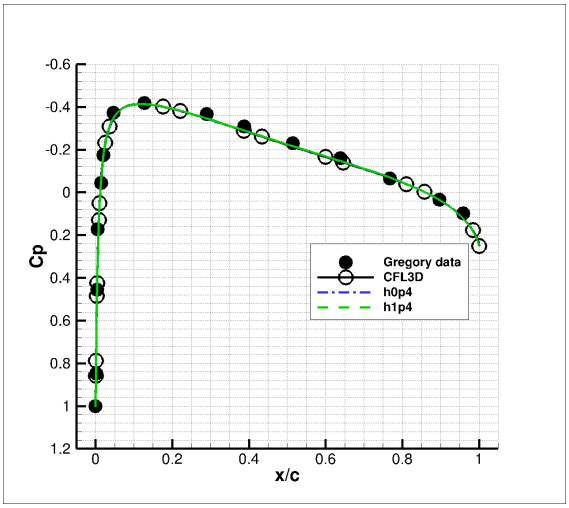
(a) $p = 1$



(b) $p = 2$

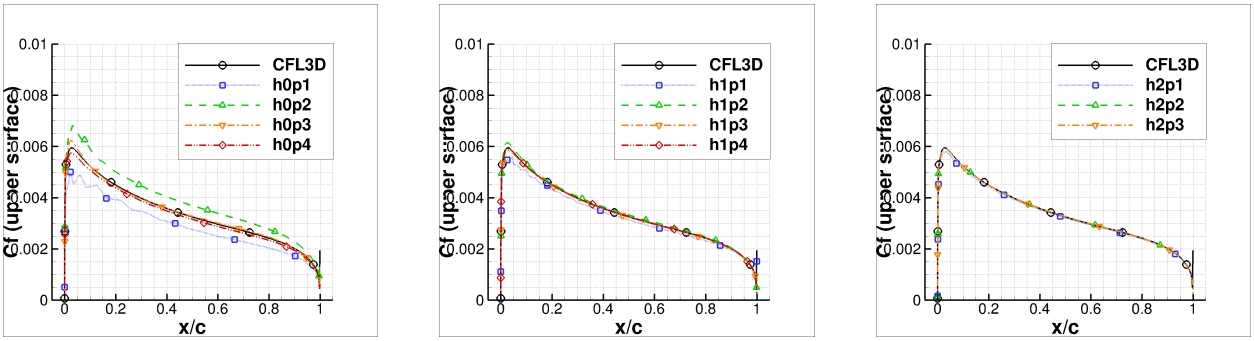


(c) $p = 3$



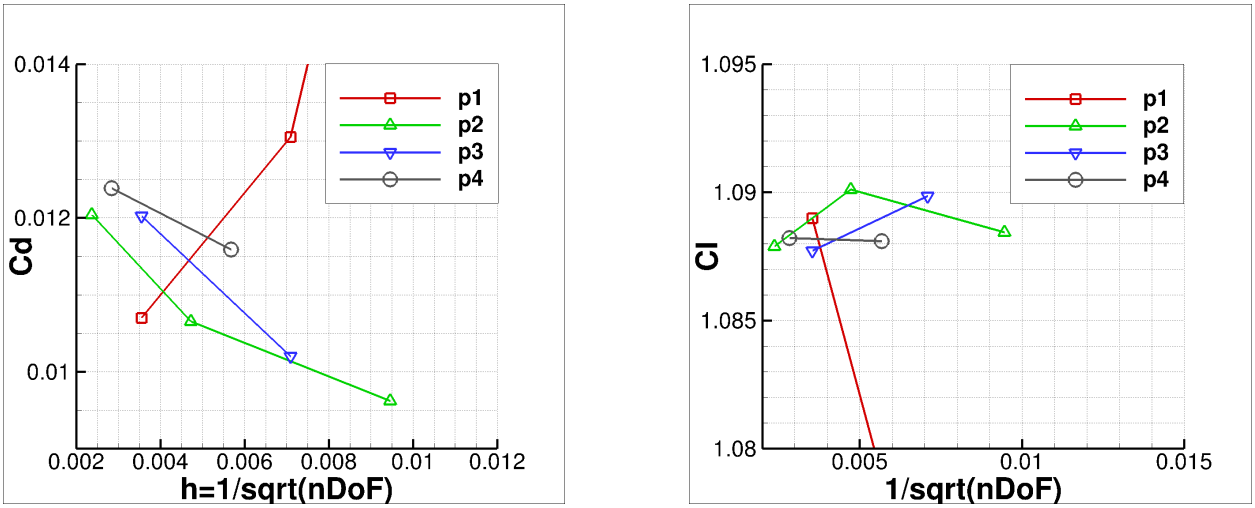
(d) $p = 4$

Figure 2.17: Surface pressure coefficient C_p for turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 0^\circ$



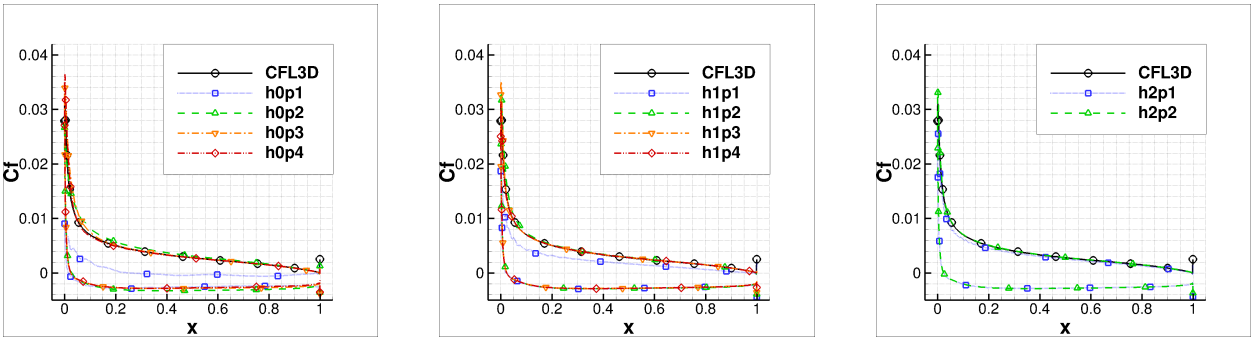
(a) $h = 0$ (b) $h = 1$ (c) $h = 2$

Figure 2.18: Skin friction coefficient C_f for turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 0^\circ$



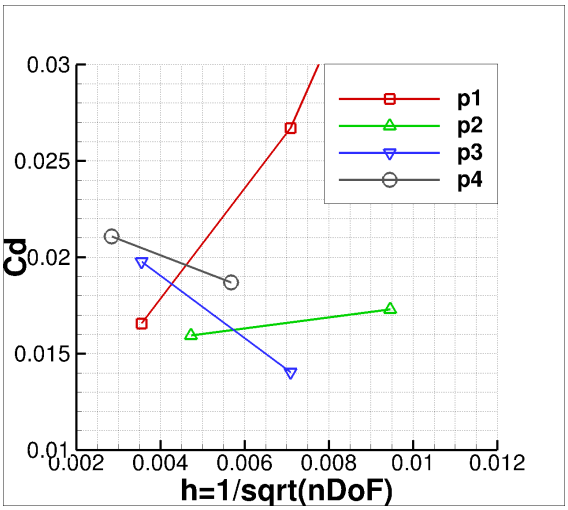
(a) the drag coefficient C_d (b) the lift coefficient C_l

Figure 2.19: Turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 10^\circ$

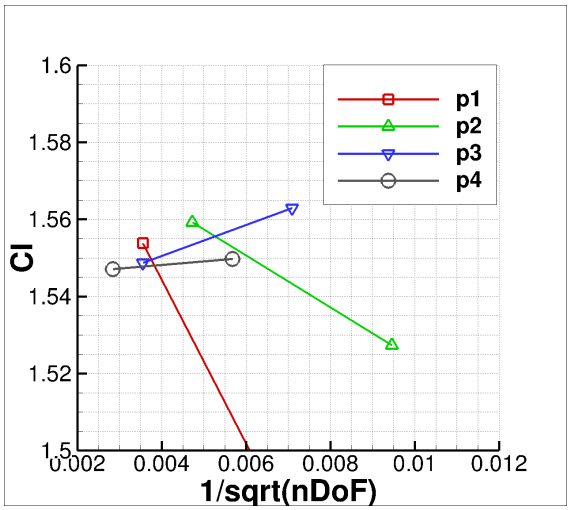


(a) $h = 0$ (b) $h = 1$ (c) $h = 2$

Figure 2.20: Skin friction coefficient C_f for turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 10^\circ$

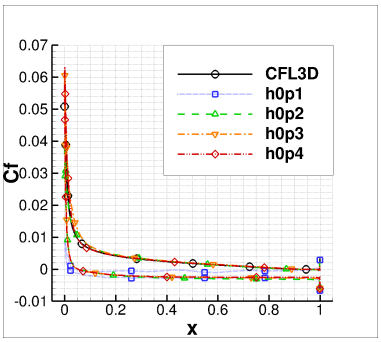


(a) the drag coefficient C_d

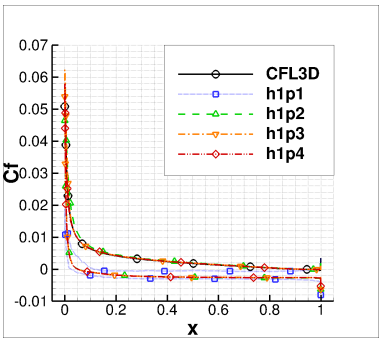


(b) the lift coefficient C_l

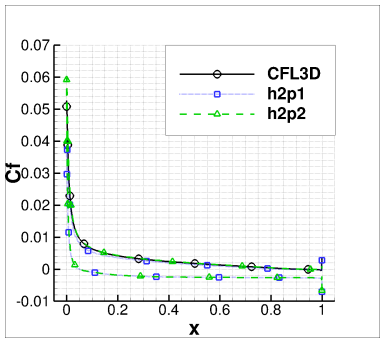
Figure 2.21: Turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 15^\circ$



(a) $h = 0$



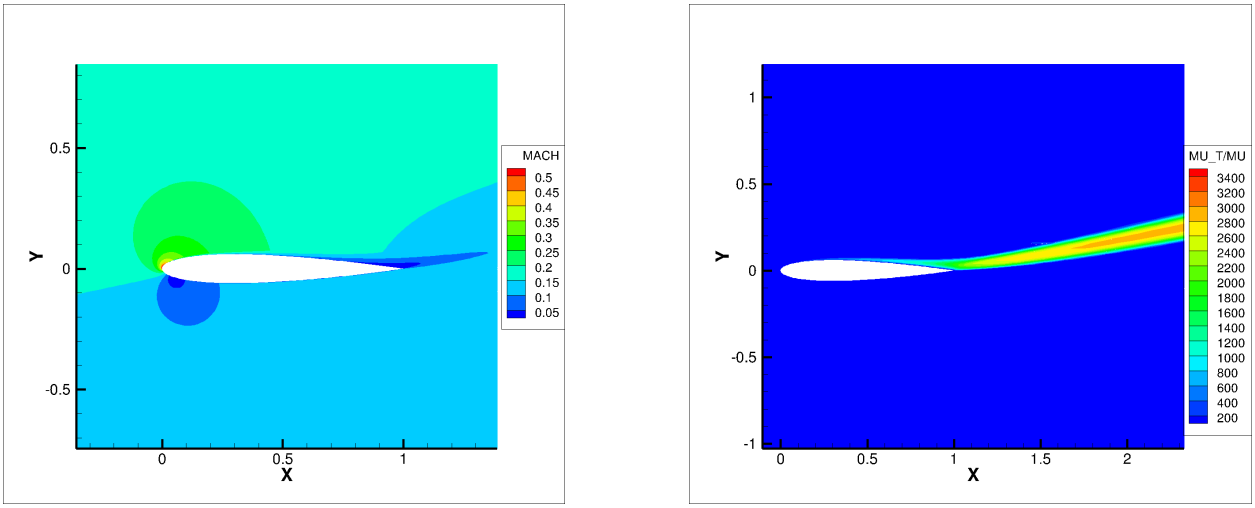
(b) $h = 1$



(c) $h = 2$

Figure 2.22: Skin friction coefficient C_f for turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 15^\circ$

Figure 2.23 displays the Mach number and non-dimensional eddy viscosity contours around the airfoil solved with the $p = 4$ discretization. Note that a very smooth solution is obtained on this medium mesh $h = 1$. At the front portion of the airfoil, the turbulent boundary layer is very thin, as shown in Figure 2.23(b).



(a) the Mach number

(b) the non-dimensional eddy viscosity

Figure 2.23: Turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 15^\circ$

2.5 Summary of Chapter

In this Chapter, we developed a CPR discretization of the RANS equations with the modified SA model. In this model, the non-dimensional length scale depends on the distance of each solution point to the nearest curved polynomial wall boundaries, which can be solved using the Eikonal equation. The high-order CPR discretization was developed to solve the Eikonal equation efficiently and robustly. We considered four test cases: the square case surrounded by wall boundaries demonstrates the capability of the CPR Eikonal solver on internal flow geometries; The super convergence given by the ring case shows the accuracy and efficiency of the proposed high-order solver; The smooth high-order wall distance solutions of the RAE2282 airfoil and the high-lift 30P30N multi-element configuration further demonstrate the robustness of the proposed solver.

Additionally, the high-order CPR discretization of the RANS equations was applied to two

benchmark test cases on the NASA TMR [2] website: turbulent flows over a flat plate and the NACA0012 airfoil. The computed high-order solutions are compared to the experimental data, the CFL3D and FUN3D results. The converged drag coefficient of the flat plate case agrees with CFL3D and FUN3D results within 0.1 count, and the Richardson extrapolation of the drag coefficient numerically verify the order of accuracy of the high-order CPR RANS-SA solver. In terms of the skin friction coefficient on the flat plate and the NACA0012 airfoil, the mesh and order independent results are obtained and show an excellent agreement with the other benchmark simulations.

However, further research remains in the mesh and order independence of the drag and lift coefficients for the NACA0012 airfoil test case, especially at high angles of attack. The adaptive mesh refinement method discussed in Chapter 4 will try to address this problem.

Chapter 3

Time Integration

3.1 Pseudo-Transient Continuation

The time discretization to Eq. (2.19) can be written as,

$$\frac{d\mathbf{Q}}{dt} = \mathbf{R}(\mathbf{Q}), \tag{3.1}$$

Only steady-state simuations are performed in this work, hence the first order backward Euler scheme is adopted

$$\frac{\mathbf{Q}^{n+1} - \mathbf{Q}^{n}}{\Delta t} = \mathbf{R}(\mathbf{Q}^{n+1}), \tag{3.2}$$

Let $\Delta\mathbf{Q} = \mathbf{Q}^{n+1} - \mathbf{Q}^{n}$ and linearize the residual using Taylor expansion

$$\mathbf{R}(\mathbf{Q}^{n+1}) \approx \mathbf{R}(\mathbf{Q}^{n}) + \frac{\partial \mathbf{R}}{\partial \mathbf{Q}}\Delta\mathbf{Q}, \tag{3.3}$$

Then the linearized version of Eq. (3.2) is

$$A\Delta\mathbf{Q} = \mathbf{R}(\mathbf{Q}^{n}), \tag{3.4}$$

where $A = I/\Delta t + \partial \mathbf{R}/\partial \mathbf{Q}$. At each cell c, the fully linearized equation can be written as

$$(\frac{I}{\Delta t} - \frac{\partial R_c}{\partial Q_c})\Delta Q_c - \sum_{nb \neq c} \frac{\partial R_c}{\partial Q_{nb}} \Delta Q_{nb} = R_c(Q^n), \tag{3.5}$$

where nb denotes all the neighboring cells contributing to the residual of cell c.

3.1.1 BLU-SGS Implicit Solver

Since storing all the implicit Jacobian matrices in the algebraic system is expensive, we employ a LU-SGS scheme to solve Eq. (3.5), i.e., we use the most recent solution for the neighboring cells Q^*_{nb},

$$(\frac{I}{\Delta t} - \frac{\partial R_c}{\partial Q_c})\Delta Q_c^{(k+1)} = R_c(Q^n) + \sum_{nb \neq c} \frac{\partial R_c}{\partial Q_{nb}} \Delta Q^*_{nb}, \tag{3.6}$$

The matrix

$$D = (\frac{I}{\Delta t} - \frac{\partial R_c}{\partial Q_c}), \tag{3.7}$$

is the element Jacobian matrix. Eq. (3.6) is then solved with an exact LU decomposition solver. To avoid storing the Jacobian matrices for the neighboring cells $\frac{\partial R_c}{\partial Q_{nb}}$, we further introduce inner iteration.

Let $\Delta^* Q_c^{(k+1)} = \Delta Q_c^{(k+1)} - \Delta Q_c^*$,Eq. (3.6) gives a very compact form of the non-linear BLUSGS

$$(\frac{I}{\Delta t} - \frac{\partial R_c}{\partial Q_c})\Delta^* Q_c^{(k+1)} = R_c(Q^*) - \frac{\Delta Q_c^*}{\Delta t}, \tag{3.8}$$

Eq. (3.8) is then solved through numbers of symmetric forward and backward sweeps with a prescribed tolerance ε for convergence. Note that at each time step, if Eq. (3.8) is solved to machine zero, the unsteady residual $R_c(Q) - \frac{\Delta Q_c}{\Delta t}$ is zero. The initial guess for Q_c^{n+1} then can be set to Q_c^n. Consequently, the initial unsteady residual is the same as the steady residual at the last time step. Then we can monitor the unsteady residual for convergence. This also indicates that, it is not necessary to solve the unsteady residual to machine zero for steady state problems. In fact, it can be more efficient to set criteria for maximum number of sweeps for Eq. (3.8), and move to

the next time step.

It is very easy to implement this numerical approach, but the computational cost is expensive because all the changes of the degrees of freedom need to be considered. To further improve the computational efficiency, the element matrices D are frozen for many intervals of time steps in this study.

3.1.2 GMRES Solver with BLU-SGS Preconditioner

In Eq. (3.4), the left-hand side (LHS) implicit operator is a very large sparse block matrix, of which the direct inversion is usually very expensive. Based on the fact that all the operations on matrix A in the GMRES is associated with the computation of matrix-vector products , which can be approximated using finite difference , the GMRES approach can be implemented without forming the Jacobian matrix explicitly or storing A. Various matrix-free implementations of GMRES have been developed[60]. Any successful GMRES iterative approach with fast convergence needs a good pre-conditioner P, which should be "close" to the implicit operator A, but much easier to invert. An equivalent preconditioned form of Eq. (3.4) is

$$P^{-1}A\Delta\mathbf{Q} = P^{-1}\mathbf{R}(\mathbf{Q}^n), \tag{3.9}$$

Note that the pre-conditioning matrix P must be formed and stored. The efficient LU-SGS preconditioner was demonstrated very good convergence properties for high-order DG schemes with polynomial degree to three[60] . In this study, BLU-SGS approach is used as the preconditioner, which means P is solved following Eq. (3.8).

3.1.3 GMRES Solver within PETSc Coupled with Different Preconditioners

PETSc integrates series of components in a highly efficient way, including low-level distributed data structures for meshes, vectors, and matrices and high-level linear, nonlinear, and times stepping solvers. The algorithmic source code is written in high-level abstractions so that it can be

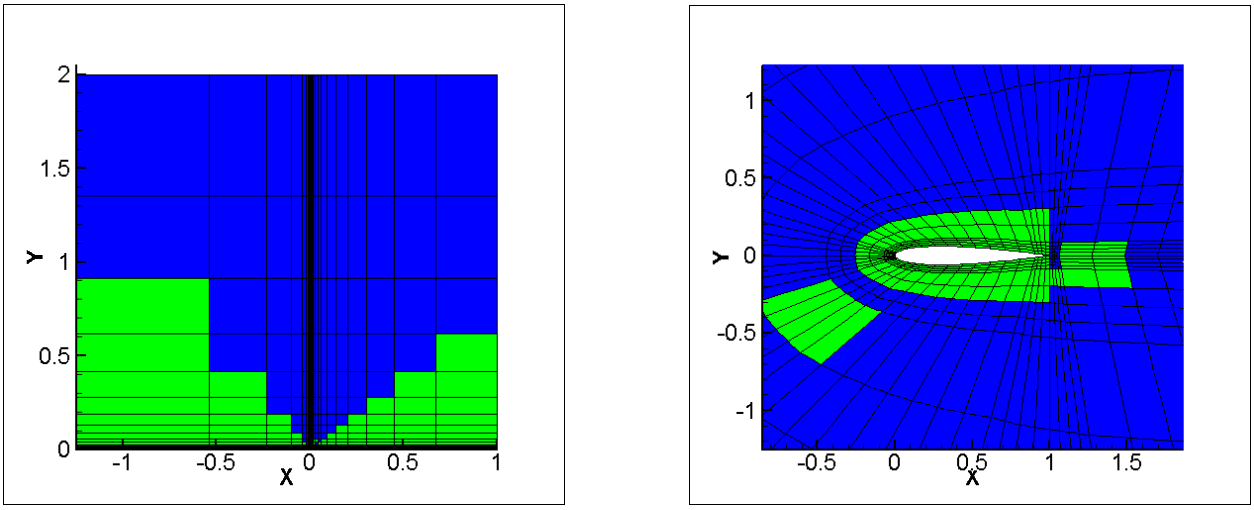
easily understood and modified. This hierarchy makes code reuse possible and promotes code flexibility. The Newton-based methods within PETSc are written in a data-structure-neutral form that uses abstractions for vectors, matrices, and linear solvers[39]. In the present study, the PETSc library is applied as the linear equation solver with the CPR framework. The GMRES solver available in PETSc is compared with the existing GMRES approach with the BLU-SGS preconditioner. Only residual evaluations and Jacobian at given state vectors in the user routines are called for the subroutines with PETSc library.

At first, numerical experiments indicate that for the laminar boundary layer on a flat plate, neither BLU-SGS nor GMRES-BLUSGS (storing main block diagonal matrix only) can achieve convergence. To remedy this problem, PETSc is directly applied to solve the non-linear system of fully linearized equations Eq. (3.4), which demonstrates a fast convergence. However, the computational cost of this approach is expensive because all the changes of the degrees of freedom need to be considered for every time step. To further improve the computational efficiency, the implicit operator matrix is frozen for several time steps in this study.

Thanks to the flexibility of PETSc, three different preconditioned GMRES within PETSc are compared through solving Eq. (3.4), namely, Jacobi (PGMRES-JACOBI), incomplete lower-upper decomposition(PGMRES-ILU), and line preconditioner (PGMRES-LINE), which are shown to be the most appropriate preconditioners for different compressible flows. We remark that the line preconditioner is based on user routines that are called by PETSc subroutines rather than PETSc built-in preconditioners such as JACOBI and ILU. Only $P^{-1}x$ is needed in PETSc, which yields to a matrix-free pre-conditioner. The lines in the present study are generated based solely on the mesh aspect-ratio (AR) to counter the stiffness associated with high AR elements. Fig. (3.1) illustrate the lines for several meshes, which are marked in green.

3.1.4 Memory Requirement

In the Jacobian matrix, the number of entries for each cell is $n_{dof} \times n_{dof}$, where n_{dof} is the number of degree of freedoms in one cell. For both BLU-SGS and GMRES- BLUSGS, only the



(a) Flat plate with 560 cells

(b) NACA0012 airfoil with 200 cells

Figure 3.1: Lines of the line preconditioner

main block diagonal matrices are stored.

$$Memory(BLU-SGS) = nElems \times Memory(cell)$$

$$Memory(cell) = n_{dof} \times n_{dof}$$

However, the GMRES solver needs extra memory to store the restart Krylov space vectors. In the present study, k = 30 for inviscid flow and k = 90 for viscous flow, which almost doubles the memory requirement (e.g. $n_{dof} = 64$ for $p = 3$ scheme on quadrilateral mesh).

$$Memory(GMRES-LUSGS) = nElems \times Memory(cell) + k \times nElems \times n_{dof}$$

$$\frac{Memory(GMRES-LUSGS)}{Memory(BLUSGS)} = 1 + \frac{k}{n_{dof}}$$

The preconditioned GMRES approach within PETSc proposed here is solving the non-linear system of fully linearized equations Eq. (3.5), which means the sparse Jacobian matrix including all neighbors is stored completely. Thus, depending on the considered element type, we get the overall memory storage (assuming large total number of elements compared to the boundary elements) as

$$Memory(PGMRES) = nElems \times Memory(cell) \times (1 + nSides) + k \times nElems \times n_{dof}$$

where $nSides$ is the number of sides for the element type (e.g. $nSides = 4$ for the quadrilateral cell). It is clear that, solving the full Jacobian matrix drastically amplifies the memory cost (excluding the cost for the preconditioner).

$$\frac{Memory(PGMRES)}{Memory(BLUSGS)} = 1 + nSides + \frac{k}{n_{dof}}$$

3.2 Line-Search for Solution Update

Line-searches are originally used in optimization problems to find a step-size along a descent direction that sufficiently reduces the value of the objective function and its gradient. When solving systems of nonlinear equations, such as Navier-Stokes equations, the line-search algorithm was developed by choosing the 2-norm of the unsteady residual as the objective function[21, 52] .

At the n_{th} time step, an under-relaxation parameter ω^n is used to ensure a physical solution update,

$$\mathbf{Q}^{n+1} = \mathbf{Q}^n + \omega^n \triangle \mathbf{Q}^n, \; such\; that\; \rho^{n+1} > 0, \; p^{n+1} > 0. \tag{3.10}$$

To determine the value of ω^n, we limit the changes in thermodynamic variables (namely, density ρ and pressure p) by $\eta = 10\%$ and require a drop in the 2-norm of the unsteady residual,

$$|\frac{\rho^{n+1} - \rho^n}{\rho^n}|^\infty < \eta, \; |\frac{p^{n+1} - p^n}{p^n}|^\infty < \eta, \tag{3.11}$$

$$||\frac{\omega^n \triangle \mathbf{Q}^n}{\triangle t} + \mathbf{R}(\mathbf{Q}^n + \omega^n \triangle \mathbf{Q}^n)|| - ||\mathbf{R}(\mathbf{Q}^n)|| < 0. \tag{3.12}$$

More details about the line-search algorithm refer to Ref. [21]. In the present work, we separate the 2-norm of residual and require a drop in the 2-norm of each conservation equation, except the last SA model equation. This reduces the effect of badly scaled discrete systems that cause the residual norm to be dominated by the worst residual component.

Furthermore, a safety check is performed based on ω^n before proceeding to the next time step. Specially, a "safe" state $\mathbf{Q}_{safe}$ is stored if $\omega^n = 1$. The solution update is summarized as

$$\mathbf{Q}^{n+1} = \begin{cases} \mathbf{Q}_{safe}, & \omega^n < 0.1 \\ \mathbf{Q}^n + \omega^n \triangle \mathbf{Q}^n, & 0.1 < \omega^n < 1 \cdot \\ \mathbf{Q}^n + \triangle \mathbf{Q}^n \; and \; \mathbf{Q}_{safe} = \mathbf{Q}^n, & \omega^n = 1 \end{cases} \tag{3.13}$$

Last, a local time stepping approach is used and the CFL evolution strategy follows Ref. [21]

$$CFL^{n+1} = \begin{cases} 0.1 \cdot CFL^n, & \omega^n < 0.1 \\ CFL^n, & 0.1 < \omega^n < 1 \cdot \\ 2 \cdot CFL^n, & \omega^n = 1 \end{cases} \tag{3.14}$$

In order to exemplify the advantage of the implemented line-search method, Fig. 3.2 displays the residual history for a subsonic turbulent flow over NACA0012 airfoil at $M_0 = 0.15$, $Re_c = 6 \times 10^6$, $\alpha = 10^\circ$ with 896 curved elements and a $p = 1$ space discretization. As expected, the line-search method saves the number of iterations by 50%.

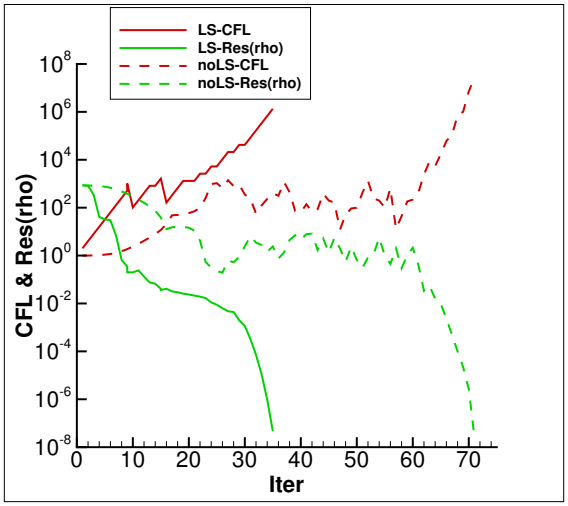


Figure 3.2: Comparison of residual convergence with and without line-search method.

3.3 Test Cases

This section presents high-order CPR solutions of steady problems in order to assess the efficiency and robustness of the proposed five time-integration schemes and preconditioners: BLU-SGS, GMRES-BLUSGS, PGMRES-JACOBI, PGMRE-ILU and PGMRES-LINE. If not stated otherwise, the following results have been obtained when the steady residual is reduced by 10 orders of magnitude.

The main aim of the first case, the flow over a flat plate, is to demonstrate that solving main block diagonal matrix only is not sufficient for boundary layer dominated flow though the memory cost is much less, while the second and third test cases, the inviscid and viscous flow over NACA0012 airfoil, have been chosen to assess the proposed five time-integration schemes and preconditioners.

For all the computations here reported the initial flow field has been set to uniform freestream conditions. The computational efficiency is compared based on converged iterations as well as work units, which was defined in the 1 st International Workshop on High-Order CFD Methods[1]. All the cases are from the Workshop.

The parameters of the restarted GMRES solver were set to 30 Krylov space vectors for inviscid flow and 90 for viscous flow, 200 maximum iterations and 10^{-2} relative convergence tolerance. The initial time step used is the largest time step that ensuring stability for each case. The growth of the time step is related to the relative residual according to the following relations:

$$\begin{aligned} dt &= min(\frac{dt_0}{f^\beta}, dt_{max}) \\ f &= \frac{\|R(Q^n)\|_{L2}}{\|R(Q^0)\|_{L2}} \\ dt_{max} &= 1.e20, and \beta = 1.05 \end{aligned} \tag{3.15}$$

where dt0 is the initial largest time step ensuring stability. For computation efficiency, the time step was increased to infinity (1.e20 in present study) after relative residual drops 3 orders magnitude.

3.3.1 Flat Plate, $M_0 = 0.5$,$Re_c = 1 \times 10^6$

The laminar boundary layer on a flat plate is simulated on both a coarse mesh $h1 = 560$ cells and a finer mesh $h2 = 2240$ cells with either 4th-order or 5th-order space discretization. Both meshes are standard meshed used in the 1st International Workshop on High-Order CFD Methods. Since this simulation can achieve full convergence (10 order residual reduction) only through solving the full linearized system, Fig. (3.3) displays the skin friction coefficient using PGMRES-LINE as the

time-integration scheme. Comparing with the h1p3 result, simulations with higher order h1p4 and finer mesh h2p3 achieve a converged result.

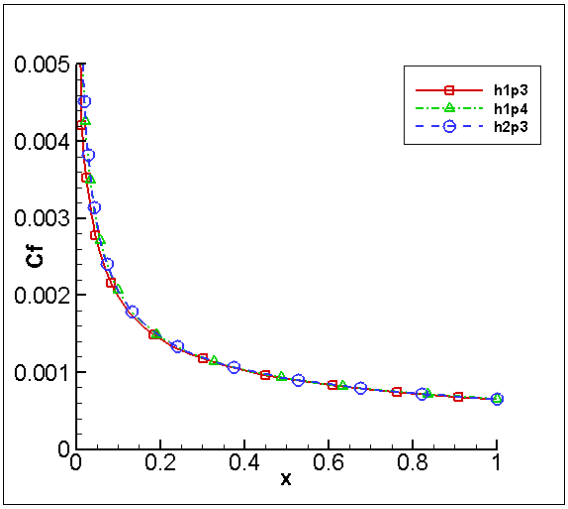


Figure 3.3: The skin friction coefficient of PGMRES-LINE for laminar flow on a flat plate

Several other schemes are compared in Table. (3.1) to Table. (3.3) in terms of work units, iterations and initial time steps to achieve convergence. “Div” denotes diverged simulation. “C-3” means the residual can only converge to 1×10^{-3} and keeps oscillating at this level. “Inf” means infinity.

Table. (3.1) shows that the simulation of the laminar boundary layer on a flat plat can converge to 1×10^{-10} only through solving the fully linearized equations. As for preconditioners, based on the 3rd-order results on different meshes Table. (3.1)and Table. (3.3), ILU(0) and line preconditioner gives the best performance for this case, while the JACOBI preconditioner cannot achieve convergence and ILU(1) takes twice as much the work units. Considering that LINE is a user-provided preconditioner and ILU is PETSc built-in preconditioner, the slightly more work units, namely 12% and 8%, can result from the optimized implementation of PETSc. Additionally, the result in Table. (3.2),which is simulated by 4th-order scheme on coarse mesh h1, demonstrates that LINE is more robust than ILU for a stiffer high-order scheme as the flow is strongly coupled in the boundary layer direction.

Schemes	Time step	Iterations	Work units	Comparison to ILU
BLUSGS	C-3			
(a)GMRES-BLUSGS	C-3			
aaPGMRES-BLUSGS	C-2			
PGMRES-ILU	Inf	7	3.803	1
PGMRES-ILU1	Inf	7	8.347	2.19
PGMRES-LINE	Inf	8	4.246	1.12

Table 3.1: Work units, iterations and initial time steps of different schemes for flow on a flat plate simulation of h1p3

Schemes	Time step	Iterations	Work units	Comparison to ILU
BLUSGS	C1			
(a)GMRES-BLUSGS	C-3			
aaPGMRES-BLUSGS	C-1			
PGMRES-ILU	C-6			
PGMRES-ILU1	DIV			
PGMRES-LINE	Inf	8	12.293	

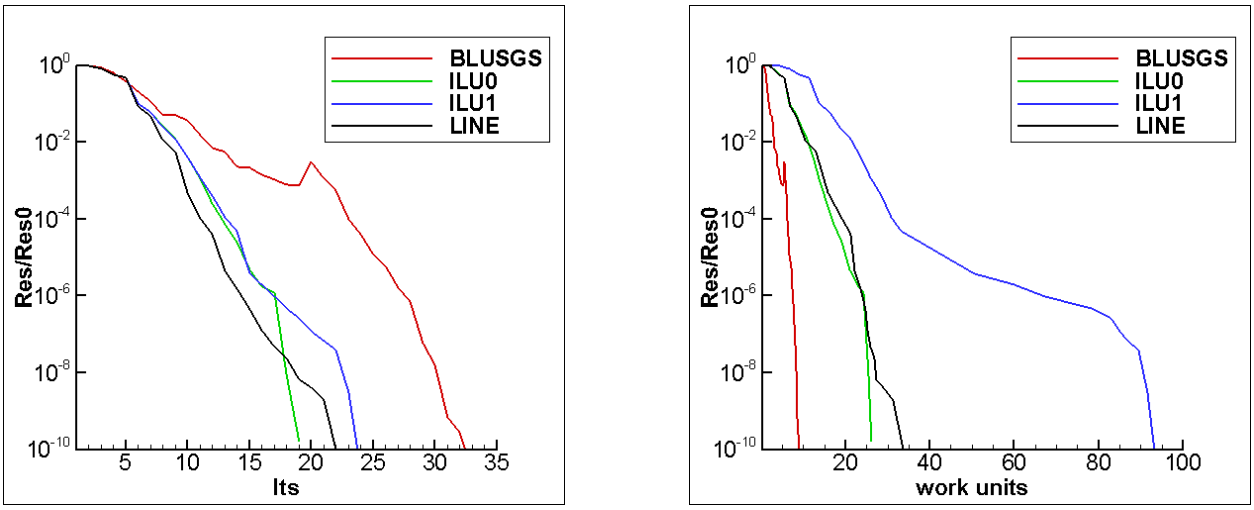
Table 3.2: Work units, iterations and initial time steps of different schemes for flow on a flat plate simulation of h1p4

Schemes	Time step	Iterations	Work units	Comparison to ILU
BLUSGS	C-2			
(a)GMRES-BLUSGS	C-3			
aaPGMRES-BLUSGS	C-1			
PGMRES-ILU	Inf	7	23.704	1
PGMRES-ILU1	Inf	7	35.657	1.50
PGMRES-LINE	Inf	8	25.742	1.08

Table 3.3: Work units, iterations and initial time steps of different schemes for flow on a flat plate simulation of h2p3

3.3.2 NACA 0012, $M_0 = 0.5, \alpha = 2^\circ$

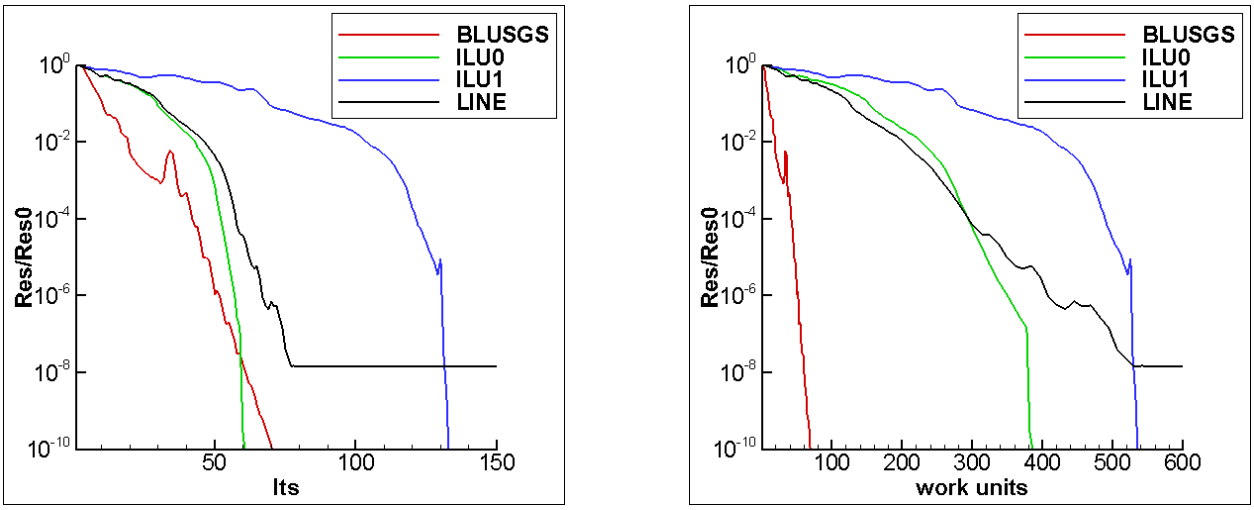
Subsonic inviscid flow over the NACA0012 airfoil is simulated on $h2 = 2240$ cells an $h3 = 8960$ cells with both 4th-order and 5th-order space discretization.The GMRES solver with BLUSGS, ILU(0), ILU(1) and LINE preconditioners are compared in Fig. (3.4) to Fig. (3.6).



(a) Iterations needed to converge

(b) Work units needed to converge

Figure 3.4: Different schemes for inviscid flow over NACA0012 airfoil with h2p3 simulation

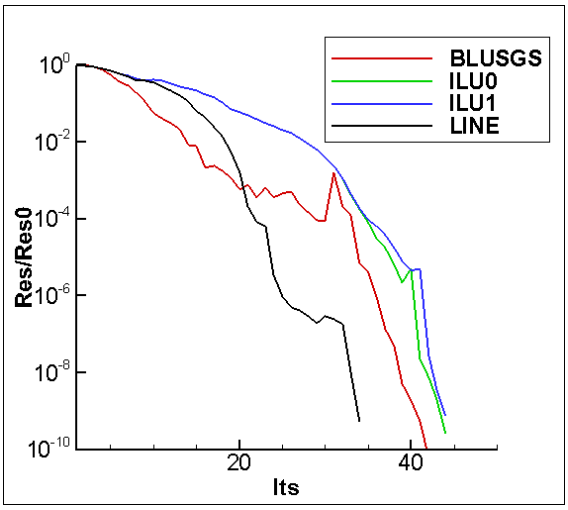


(a) Iterations needed to converge

(b) Work units needed to converge

Figure 3.5: Different schemes for inviscid flow over NACA0012 airfoil with h3p3 simulation

Fig. (3.4) to Fig. (3.6) shows that, while GMRES -BLUSGS takes more iterations to converge, it is still the most efficient scheme for this case in work units. GMRES solver within PETSc solving



(a) Iterations needed to converge

(b) Work units needed to converge

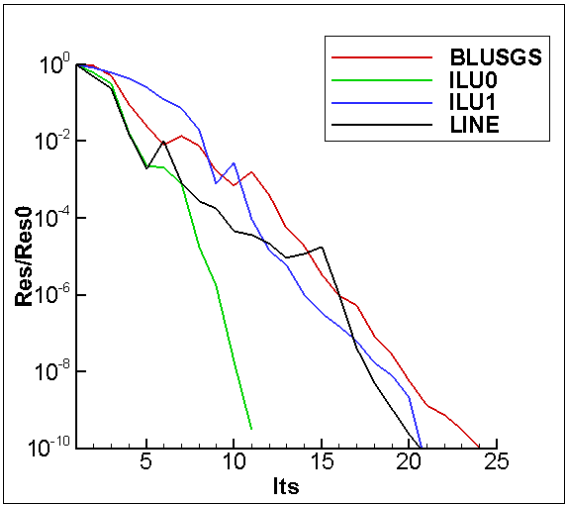
Figure 3.6: Different schemes for inviscid flow over NACA0012 airfoil with h2p4 simulation

the full linearized system does not perform as well as the GMRES -BLUSGS, and the more the total degree of freedoms the worse it performs since more off-diagonal entries in the Jacobian matrix need to be stored and solved. Among the three different preconditioners coupled with PGMRES, ILU and LINE give the best performance and LINE is more robust for higher-order scheme, which agrees with the conclusion of flat plate test. Also, ILU(1) only amplifies work units in the reason of keeping more entries during decomposition, while does not improve robustness in sense of the same or even more iterations than ILU.

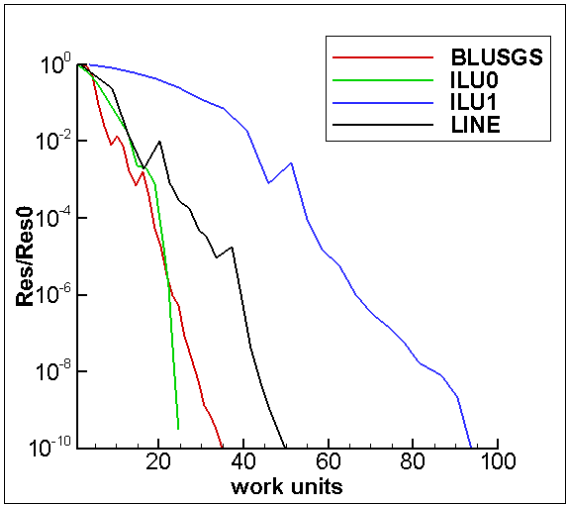
3.3.3 NACA 0012, $M_0 = 0.5, Re_c = 5 \times 10^3, \alpha = 1^\circ$

Subsonic viscous flow over the NACA0012 airfoil is simulated on both a coarse mesh of $h2 = 2240$ cells and a finer mesh of $h3 = 8960$ cells with with both 4th-order and 5th-order space discretization.The GMRES solver with BLUSGS, ILU(0), ILU(1) and LINE preconditioners are compared in Fig. (3.7) to Fig. (3.9).

Fig. (3.4) to Fig. (3.6) shows that PGMRES-ILU and PGMRES-LINE demonstrates to be the most robust schemes with larger initial time step, less iterations and competitive work units. Since GMRES-BLUSGS requires much less memory cost and comparable CPU time, it is recommended

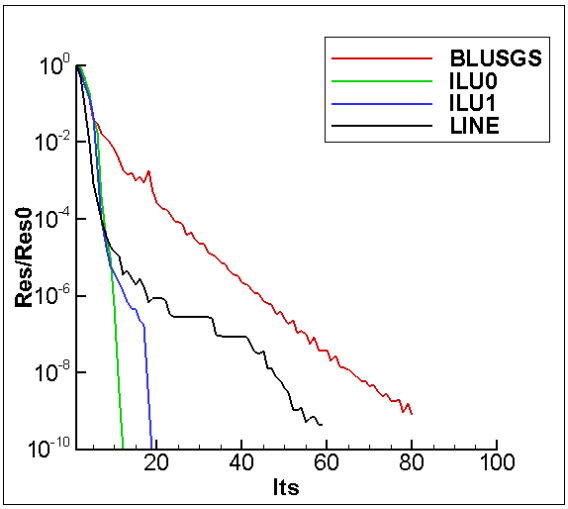


(a) Iterations needed to converge

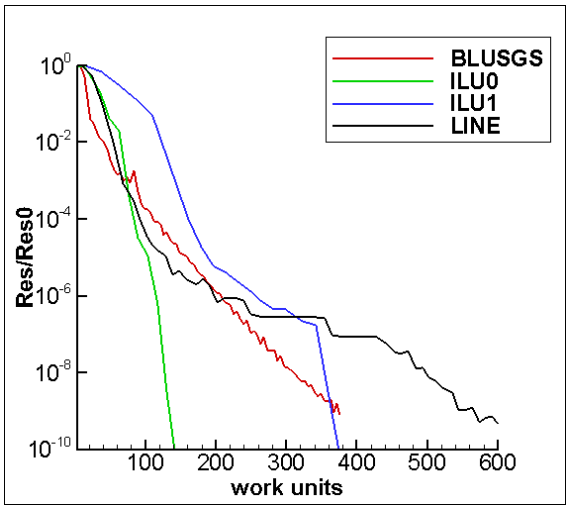


(b) Work units needed to converge

Figure 3.7: Different schemes for viscous flow over NACA0012 airfoil with h2p3 simulation

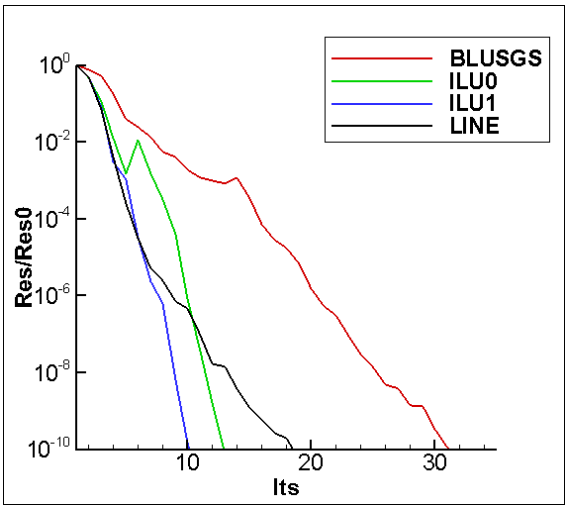


(a) Iterations needed to converge



(b) Work units needed to converge

Figure 3.8: Different schemes for viscous flow over NACA0012 airfoil with h3p3 simulation



(a) Iterations needed to converge

(b) Work units needed to converge

Figure 3.9: Different schemes for viscous flow over NACA0012 airfoil with h2p4 simulation

for memory-limited numerical simulations. Among the three different preconditioners coupled with PGMRES, given ILU shows almost the same performance as line precondition, LINE is more stable in the sense that a larger initial time step can be used for finer mesh. Requiring almost three times work units, ILU(1) is not as robust as ILU(0) and line preconditioner, which the other two test cases.

Comparing the inviscid and viscous flow over NACA0012 airfoil, GMRES-BLUSGS does not give the impressive advantage for viscous flow as that for inviscid flow in terms of CPU time. This is because that the flux gradient in viscous flow requires more information from neighboring cells and thus the Jacobian matrix is not diagonal dominant as that of inviscid flow, solving the full linearized system is more reasonable and obviously accurate than only storing the main diagonal block matrix, which also explains the conclusion in flat plate case, that the simulation of laminar flow over a flat plate can achieve full convergence only through solving the full linearized system.

3.4 Summary of Chapter

In this Chapter, a comparative study of several well-known or recently developed implicit time integration schemes with the CPR discretization is carried out. The BLU-SGS approach,

the matrix-free GMRES with BLU-SGS as a preconditioner, and the GMRES solver available in PETSc with three different preconditioners (JACOBI, ILU, LINE) are adopted in the development. The BLUSGS scheme and GMRES approach with BLU-SGS as precondition only store the main diagonal matrix to save memory cost. To achieve convergence for the laminar boundary layer on a flat plate, the fully linearized system is solved with the GMRES solver available in PETSc. The proposed time-integration schemes are tested for several cases described in Section I. For inviscid and viscous flow over NACA0012 airfoil, GMRES with BLU-SGS as the preconditioner is demonstrated the most efficient and economical. However, in sense of robustness, GMRES solver solving the fully linearized system is recommended for viscous flow. Considering the convergence for laminar flow over a flat plate, LINE preconditioned GMRES solver solving the fully linearized system is the most efficient, robust and general scheme for viscous flow.

Chapter 4

Error Estimation and Mesh Adaptation

4.1 Adjoint-based Error Estimation

Adjoint-based error estimation relates a specific functional output directly to the local residuals by the adjoint solution, which can be used to construct a very effective error indicator to drive an adaptive procedure toward any engineering output. The discrete adjoint $\tilde{\psi}_l$ is defined to relates the perturbation of the output to the perturbation of the local residual as

$$\delta \mathscr{J}_l \equiv \tilde{\psi}_l^T \delta \mathscr{R}_l \tag{4.1}$$

where $\mathscr{R}_l(Q_l)$ is the local residual of the discrete solution, and $\mathscr{J}_l(Q_l)$ is the output(i.e. Lift, Drag) of the discrete solution.

In order to solve the discrete adjoint, both sides of Eq. (4.1) are linearized by the first order approximation,

$$\delta \mathscr{J}_l = \mathscr{J}_l(Q_l + \delta Q_l) - \mathscr{J}_l(Q_l) \approx \frac{\partial \mathscr{J}_l}{\partial Q_l} \delta Q_l \tag{4.2}$$

$$\delta \mathscr{R}_l = \mathscr{R}_l(Q_l + \delta Q_l) - \mathscr{R}_l(Q_l) \approx \frac{\partial \mathscr{R}_l}{\partial Q_l} \delta Q_l \tag{4.3}$$

Substituting the linearization in Eq. (4.2) to the definition of discrete adjoint in Eq. (4.1), we obtain

the discrete adjoint equation,

$$\frac{\partial \mathscr{R}_l}{\partial Q_l}^T \tilde{\psi}_l = \frac{\partial \mathscr{J}_l}{\partial Q_l}^T \tag{4.4}$$

Following the detailed discussion about the dual-consistency of the CPR method in Ref. [65]. the discretized discrete adjoint equation for the CPR method reads

$$-\sum_i \sum_j \frac{\partial \mathscr{R}_{i,j}}{\partial Q_l} \omega_j |J_{i,j}| \psi_{i,j} = \frac{\partial \mathscr{J}}{\partial Q_l}, \tag{4.5}$$

where $\mathscr{R}_{i,j}$ and $\psi_{i,j}$ are the pointwise residual and adjoint variable defined on each solution point j of cell i arising from a CPR scheme, and ω_j and $|J_{i,j}|$ are the quadrature weight and the element Jacobian at the solution point j.

Let Q_l denotes an approximate solution to the analytic solution Q. The difference between them can be interpreted as a solution perturbation $\delta Q = Q - Q_l$. The output error defined as $\delta \mathscr{J} = \mathscr{J}(Q_l) - \mathscr{J}(Q)$ can be estimated by the adjoint weighted residual method [12],

$$\delta \mathscr{J} \approx \int_\Omega \psi(\mathscr{R}(Q_l) - \mathscr{R}(Q)) d\Omega \tag{4.6}$$

Note the discrete residual evaluated by the analytic solution should be vanished $\mathscr{R}(Q) = 0$. Then we can estimate the output error on the fine space h evaluated by the fine space solution Q_h as,

$$\delta \mathscr{J}_h(Q_h) \approx \int_\Omega \psi \mathscr{R}(Q_h) d\Omega \approx \sum_i \sum_j \omega_j |J_{i,j}| \psi^h_{i,j} \mathscr{R}_{i,j}(Q^H_h) \tag{4.7}$$

Solving Eq.4.5 respectively on the coarse solution space Q_H and finer solution space Q_h, gives the adjoint solution ψ_H and ψ_h correspondingly. The finer solution Q_h in Eq. (4.7) is approximated by enriching the degree of the solution polynomial Q_H, and performing several steps of GMRES relaxation,

$$Q^H_h = I^H_h Q_H, \tag{4.8}$$

where I^H_h denotes an injection operator.

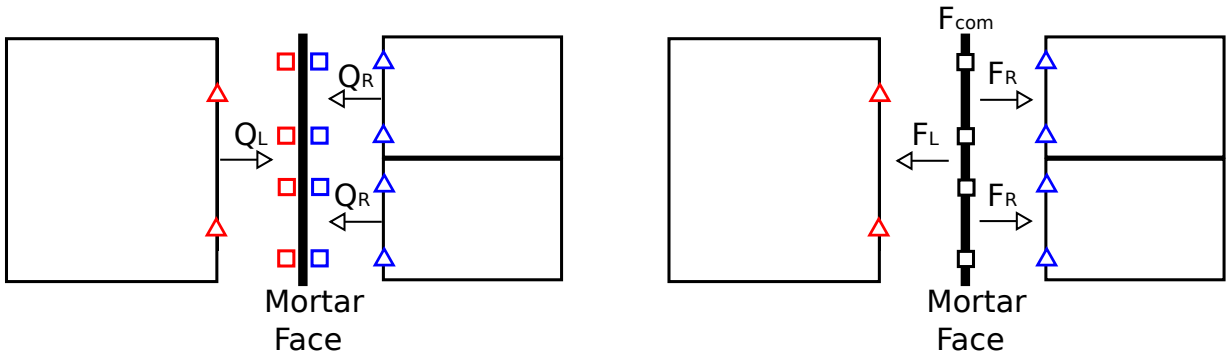
The adjoint-based local error indicator η_i used in this paper is defined by taking an absolute value of the elemental output error contribution

$$\eta_i = |\sum_j \omega_j |J_{i,j}| \left(\psi_h - I_h^H \psi_H\right)_{i,j} \mathscr{R}_{i,j}(Q_h^H)|. \tag{4.9}$$

Here, to achieve a better estimation, the adjoint defect between the coarse level and fine level $\psi_h - I_h^H \psi_H$ is used. For systems of equation, the local error indicators are formed by summing together every component's contribution to the functional error estimation.

4.2 h-adaptation and Non-conforming mesh

The error indicators defined above are used to drive a fixed-fraction isotropic h-adaptation. In this approach, a certain fraction $f = 10\%$ of the current elements with the largest local error indicators η are marked for h-refinements. Non-conforming interfaces between cells with different h levels are created in the adaptation process. In order to ensure the solution smoothness, only one level difference of h-refinement between neighboring cells is allowed. The "mortar" element method developed by Kopriva[46] is used to compute the common numerical flux on those non-conforming interfaces with hanging nodes. First, the solution from the left and right sides of the face are prolongated to the mortar surface by a simple interpolation process, see Fig. 4.1(a). Then, the common flux are computed by solving the Riemann problem on the mortar surface. The last step is to project the common flux on the mortar surface back to the original space, see Fig. 4.1(b). Here, the standard L_2 projection is utilized to preserve the average.



(a) Solution prolongation

(b) Common flux restriction

Figure 4.1: Mortar face operation for a non-conforming face ($k = 1$, $\triangle$: FPs, $\square$: DOFs on the mortar face).

Mesh refinement is performed in the original element's polynomial space using the reference coordinates, so that the refined elements inherit the same geometry approximation order. For elements on the geometry boundaries, an extra remapping process is employed to snap the boundary points to the truth geometry at each adaptation level. Fig. 4.2 depicts the procedure of the adjoint-based h-adaptation for the CPR method.

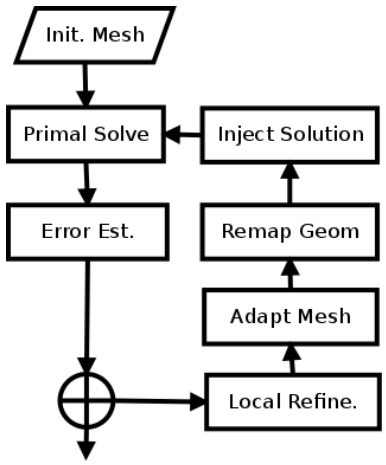


Figure 4.2: The procedure of the adjoint-based h-adaptation.

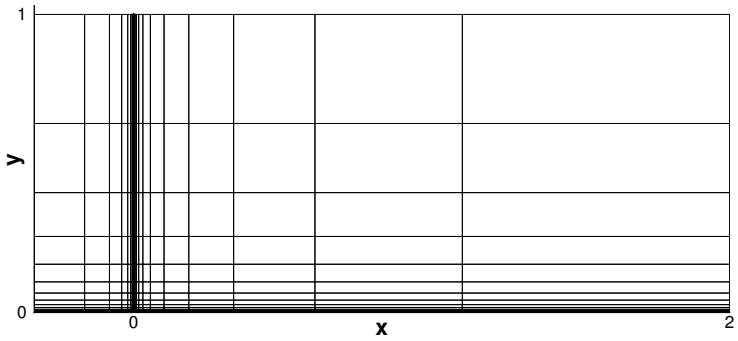
4.3 Numerical Results for CPR RANS-SA Solver

In this section, we apply the adaptive CPR discretization of RANS equations with the SA model to three steady flow problems, including two cases already discussed in Chapter 2. We will compare the efficiency and accuracy of the adaptive results with the uniform refinement results, and solve the problems left in Chapter 2: the engineering output(i.e., lift and drag coefficients) didn't achieve mesh and order independent results even with a relatively fine mesh for the turbulent flow over the NACA0012 airfoil.

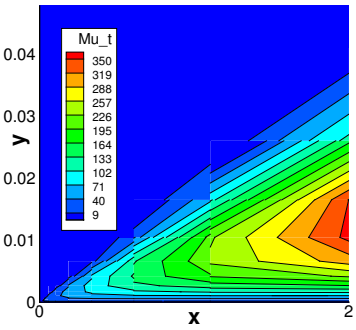
The initial conditions of all the $p = 1$ discretizations have been set to uniform freestream. For computational efficiency, each converged lower order solution is used as the initial flow field of the next higher order solution. The parameters of the restarted GMRES solver is set to 90 Krylov space vectors, 200 maximum iterations and 10^{-8} relative convergence tolerance. The initial time step is $CFL = 1$. The CFL grows following the line search method reported in Section 3.2. For computation efficiency, the CFL is increased to infinity (1.e10 in present study) after the residual drops 6 orders of magnitude. The residuals for all cases drop 10 orders of magnitude.

4.3.1 Turbulent flow over a flat plate

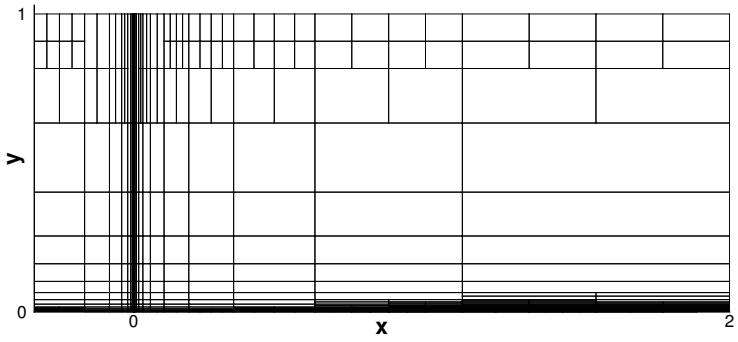
First we consider a subsonic, turbulent flow over a flat plate. This test case is from the NASA Turbulence Modeling Resource (TMR) website[2]. The simulation is performed on a rectangular domain of size $[-\frac{1}{3}, 2] \times [0, 1] \subset \mathbf{R}^2$, with the freestream Mach number $M_0 = 0.2$, and the Reynolds number $Re = 5 \times 10^6$ based on the plate length of 1. The plate spans from $x = 0.0$ to $x = 2.0$, along which the adiabatic no-slip wall boundary condition is imposed. A symmetry boundary condition is specified on the first part of the lower boundary, which results in a singularity at the leading edge of the plate, between the symmetry boundary condition and the no-slip boundary condition. The total pressure and static pressure are fixed respectively on the left and right boundaries. Farfield characteristic boundary condition is enforced on the upper surface. For the laminar viscosity, Sutherland's law with $T_s = 110K$ and $T_{ref} = 300K$ is used.



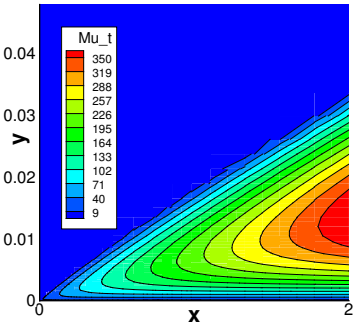
(a) Initial mesh



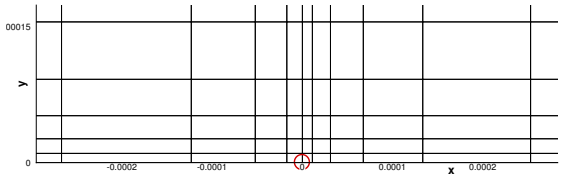
(b) Nondimensional eddy viscosity on initial mesh



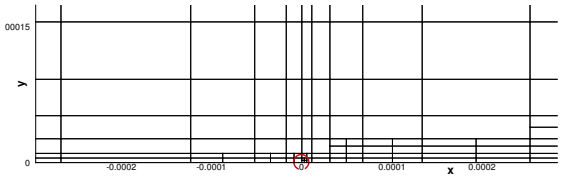
(c) Adaptive refined mesh



(d) Nondimensional eddy viscosity on refined mesh



(e) Initial mesh around leading edge



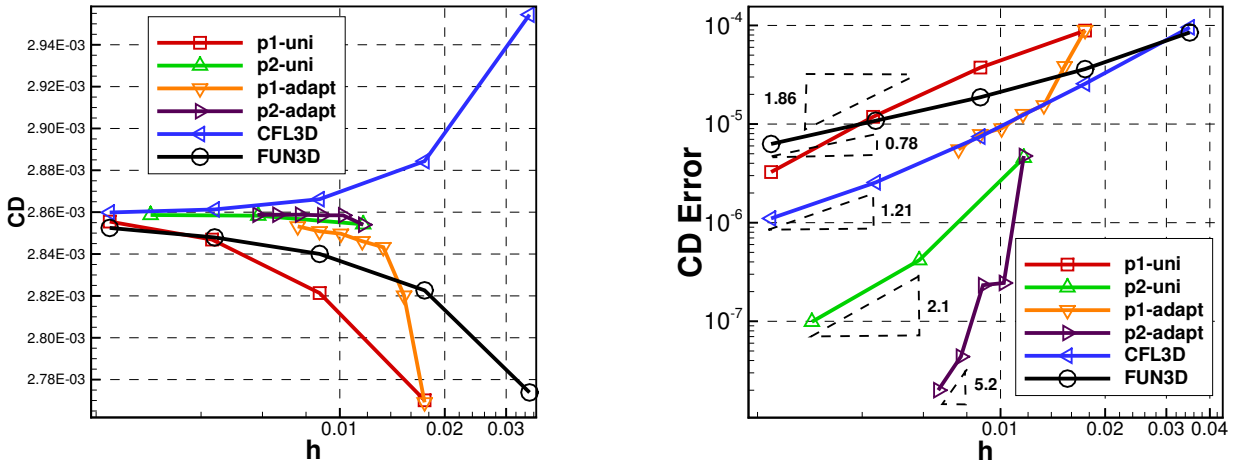
(f) Adaptive refined mesh around leading edge

Figure 4.3: The initial and adaptive results for the turbulent flow over a flat plate problem at $M_0 = 0.2, Re_c = 5 \times 10^6 (p = 2)$

The $p = 1,\ 2$ CPR discretizations with the Gauss points as the SPs/FPs are tested on four uniform refined meshes, as well as isotropic h-adaptations. The initial mesh, as shown in Figure 4.3(a), consists of 34×24 quadrilateral elements, which has an approximate average $y^+ \approx 1.7$ over the plate. In this case, the drag adjoint error indicator drives the isotropic h-adaptions. Figure 4.3 displays the adaptive refined mesh and the non-dimensional eddy viscosity contours from the finest

adaptation stage with the $p = 2$ discretization. The singularity point at the leading edge and the elements along the lower boundary are refined repeatedly on each adaptation level.

Figure 4.4(a) compares the convergence histories of the drag coefficient with CFL3D and FUN3D results[47]. The converged results agree with CFL3D and FUN3D within 0.1 count. We choose the the finest $p = 2$ adaptive result as the truth $C_D = 0.00285875$. Figure 4.4(b) compares the error of drag coefficient for all simulations, which indicates that the adaptive simulations converge much faster than the uniform refinements. With h-adaptation, effective convergence rates of 1.2 and 5.2 were achieved for $p = 1$ and $p = 2$ discretizations respectively. Figure 4.4(b) demonstrates that the adjoint based h-adaptation approach reduces the number of DOFs by orders of magnitude.



(a) the drag coefficient C_d

(b) the error of drag coefficient

Figure 4.4: C_D convergence for the turbulent flow over a flat plate at $M_0 = 0.2, Re_c = 5 \times 10^6$

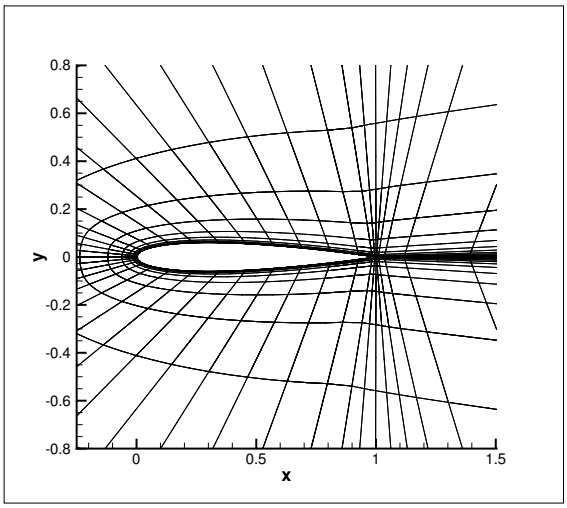
4.3.2 Turbulent flow over a NACA0012 airfoil

The next test case is a turbulent flow over the NACA0012 airfoil at Mach number $M_0 = 0.15$, Reynolds number $Re = 6 \times 10^6$ based on the airfoil chord length of 1, and angles of attack $\alpha = 0^\circ, 10^\circ$. This case is used as a validation case of CFD codes on the TMR webpage[2] , by comparing all the CFD results with the experimental results.

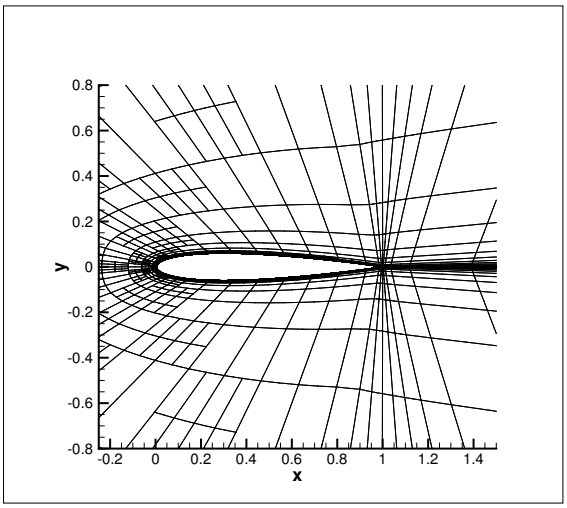
The farfield boundary is located almost 500 chords away from the airfoil. The two initial meshes for $\alpha = 0^\circ$ and $\alpha = 10^\circ$ are C-type grids with 54×23 and 56×16 fourth-order curved elements, respectively. Both of them give $y^+ \approx 10$ on the first layer grid, while the mesh for $\alpha = 10^\circ$ is more clustered at the trailing edge and boundary layer region with less grid points in the normal direction to the airfoil. Three uniform refined meshes and the isotropic h-adaptations are tested with discretization orders of $p = 1, 2, 3, 4$. The drag adjoint error indicator drives the isotropic h-adaptation. Fig. 4.5 compares Mach contours solved at angle of attack $\alpha = 0^\circ$ on the initial mesh and the adaptive refined mesh from the finest adaptation stage. Fig. 4.6 compares non-dimensional eddy viscosity solved at angle of attack $\alpha = 10^\circ$ on the initial mesh and the adaptive refined mesh. Note that regions near the stagnation streamlines and inside the boundary layer are targeted for mesh refinements. The trailing edge and wake region are also refined repeatedly to alleviate the effect of the geometry singularity. In Fig. 4.6(d) a very smooth solution is obtained through adaptations, and the turbulent boundary layer is very thin at the front portion of the airfoil.

Fig. 4.7 compares the convergence of the drag and lift coefficients for angle of attacks $\alpha = 0^\circ, 10^\circ$ with the CFL3D and FUN3D results obtained on an extremely fine mesh of 897×257 elements. The outputs from the the final stage of the adaptive simulations with the $p = 4$ discretization are chosen as the truth values to compute the error of outputs. As shown in Fig. 4.7, the adaptation results converge much faster than the results of uniform refinements, and the drag coefficients solved by $p = 3, 4$ discretizations converge to the same value between the reference values given by CFL3D and FUN3D, which demonstrates that the mesh and order independent results are obtained. In terms of the lift coefficients depicted in Fig. 4.7(e), results of uniform refinements are far away from the results of adaptation even at the very fine mesh, which indicates that the adjoint based h-adaptation approach captures the under-resolved regions and assign the number of degrees of freedom (DOFs) there, so that the functional error is efficiently reduced in terms of DOFs.

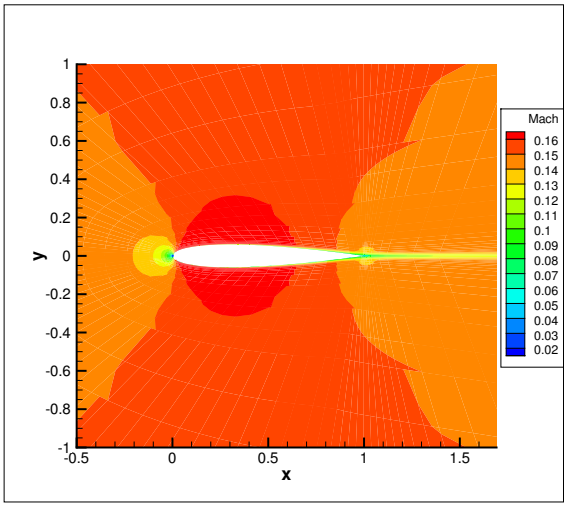
The computational results are compared with experimental data as well as the CFL3D[47] results. The CFL3D results used here are obtained with 897×257 elements and is considered as benchmark solutions for this case. Fig. 4.8 and Fig. 4.9 compare the surface pressure coefficient



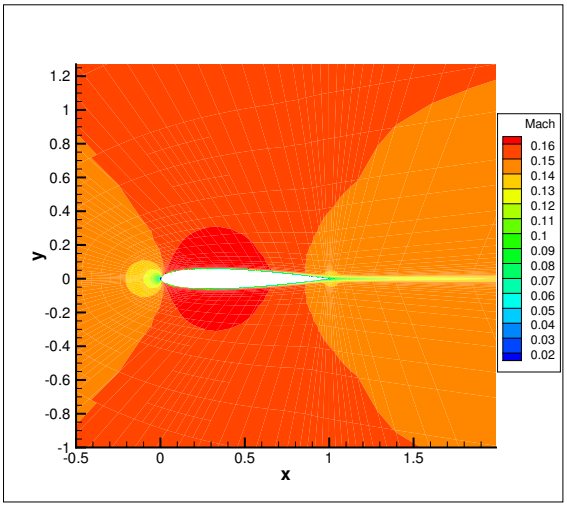
(a) The initial mesh



(b) The adaptive refined mesh

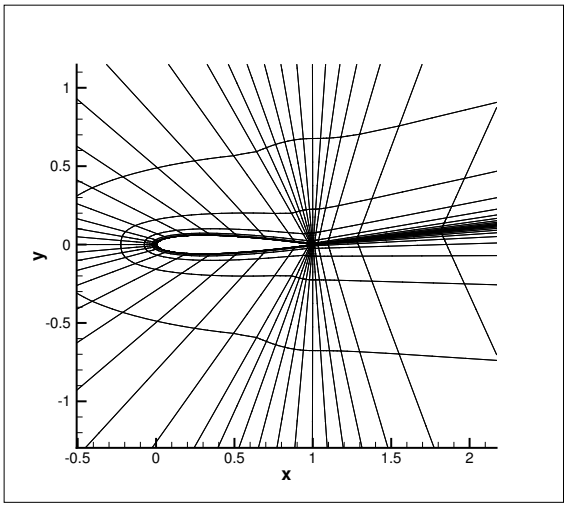


(c) Mach number contours on the initial mesh

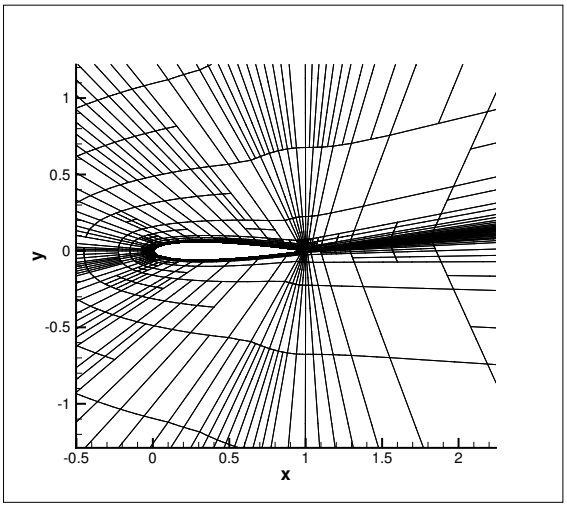


(d) Mach number contours on the adaptive refined mesh

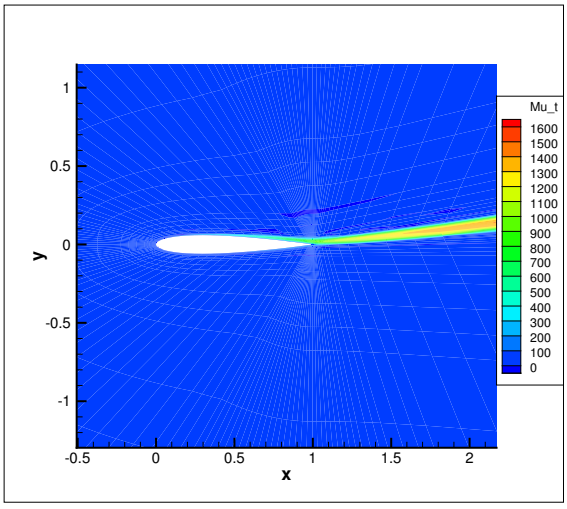
Figure 4.5: Adjoint-based h-adaptation for the NACA 0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 0^\circ, p = 2$



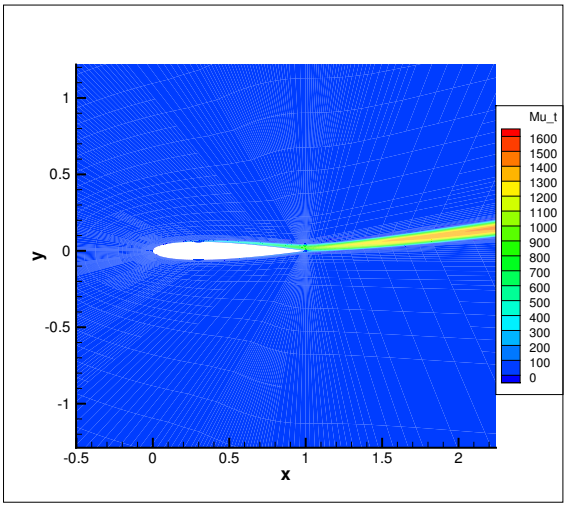
(a) The initial mesh



(b) The adaptive refined mesh

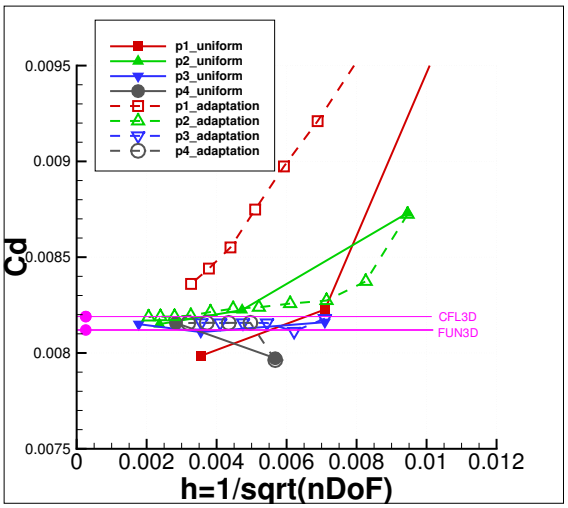


(c) Nondimensional eddy viscosity contours on the initial mesh

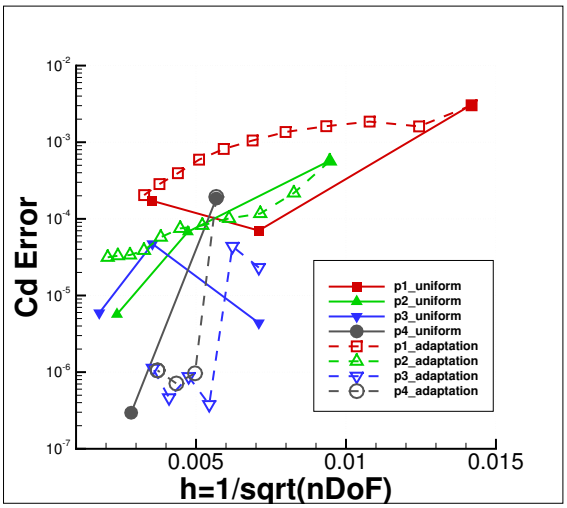


(d) Nondimensional eddy viscosity contours on the adaptive refined mesh

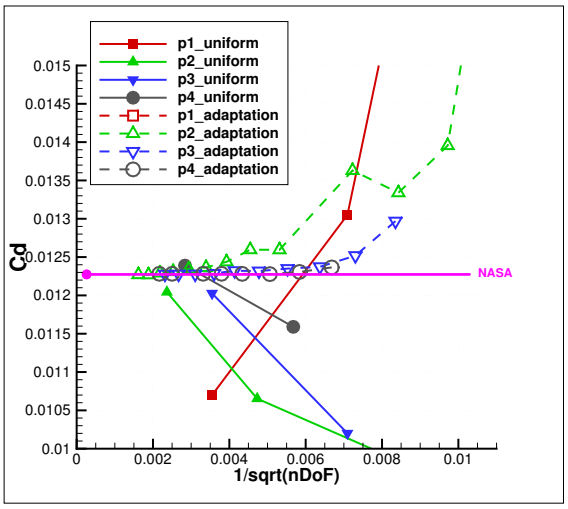
Figure 4.6: Adjoint-based h-adaptation for the NACA 0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 10^\circ, p = 3$



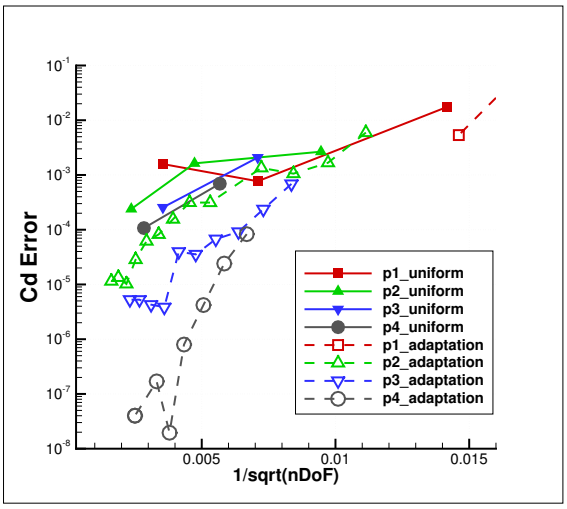
(a) Drag coefficient C_d, $\alpha = 0°$



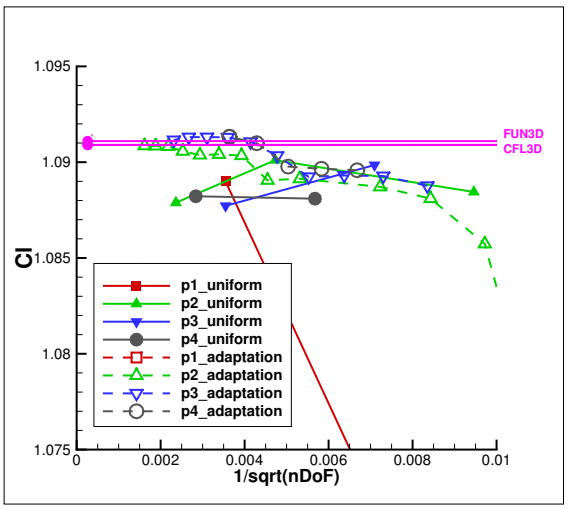
(b) Error of drag coefficient, $\alpha = 0°$



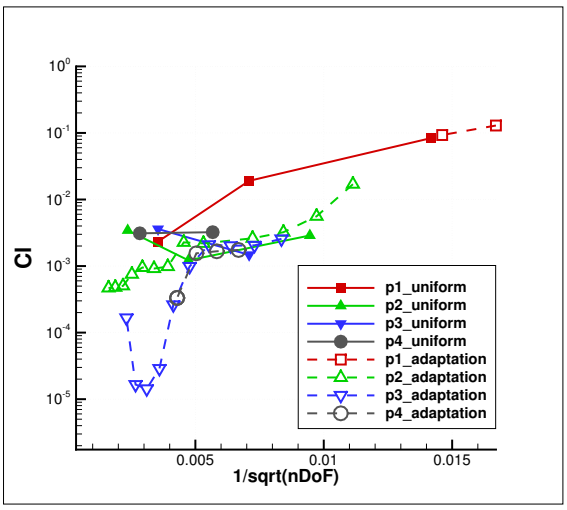
(c) Drag coefficient C_d, $\alpha = 10°$



(d) Error of drag coefficient, $\alpha = 10°$



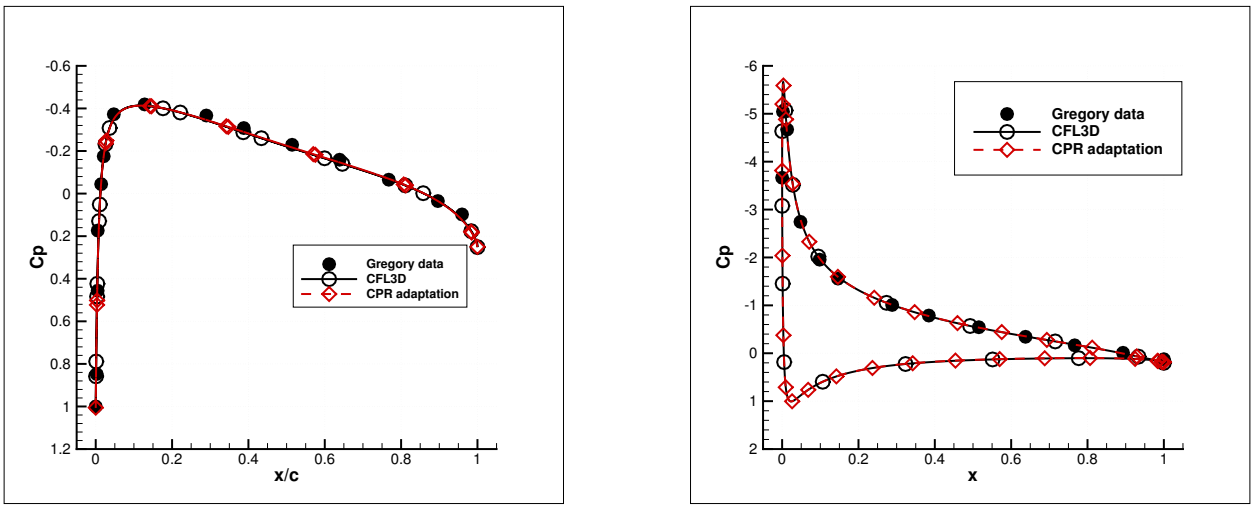
(e) Lift coefficient C_l, $\alpha = 10°$



(f) Error of lift coefficient, $\alpha = 10°$

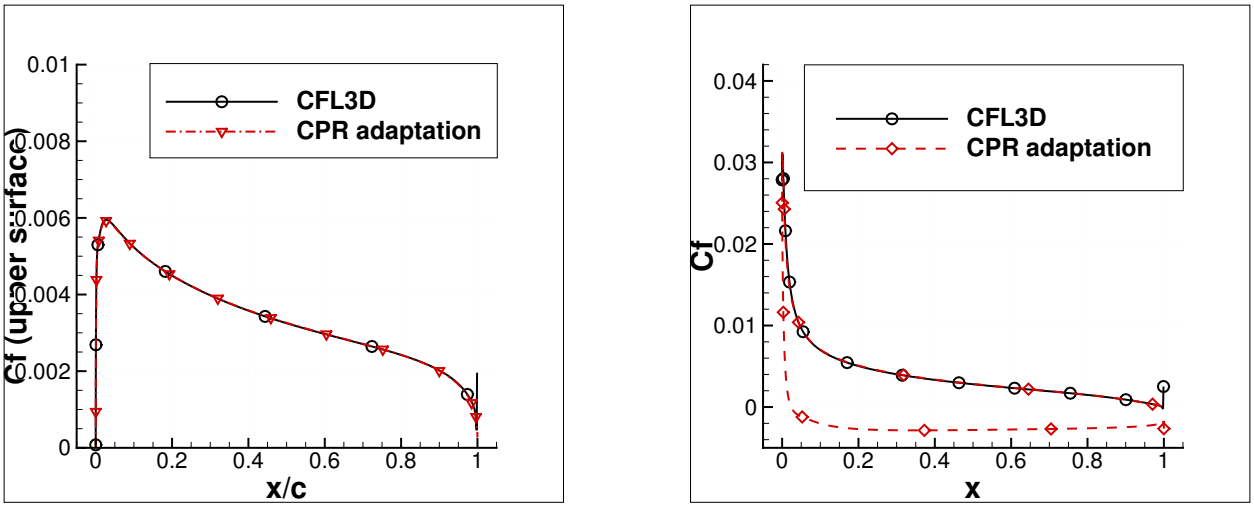
Figure 4.7: Drag and lift coefficients for turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6, \alpha = 0°, 10°$

and skin friction coefficient with the experimental data as well as the CFL3D results. The computational results are from the the final stage of the adaptive simulations with the $p = 3$ discretization. The present simulations show excellent agreement with experimental data and CFL3D results as expected.



(a) $\alpha = 0^\circ$ (b) $\alpha = 10^\circ$

Figure 4.8: Surface pressure coefficient C_p for turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6$

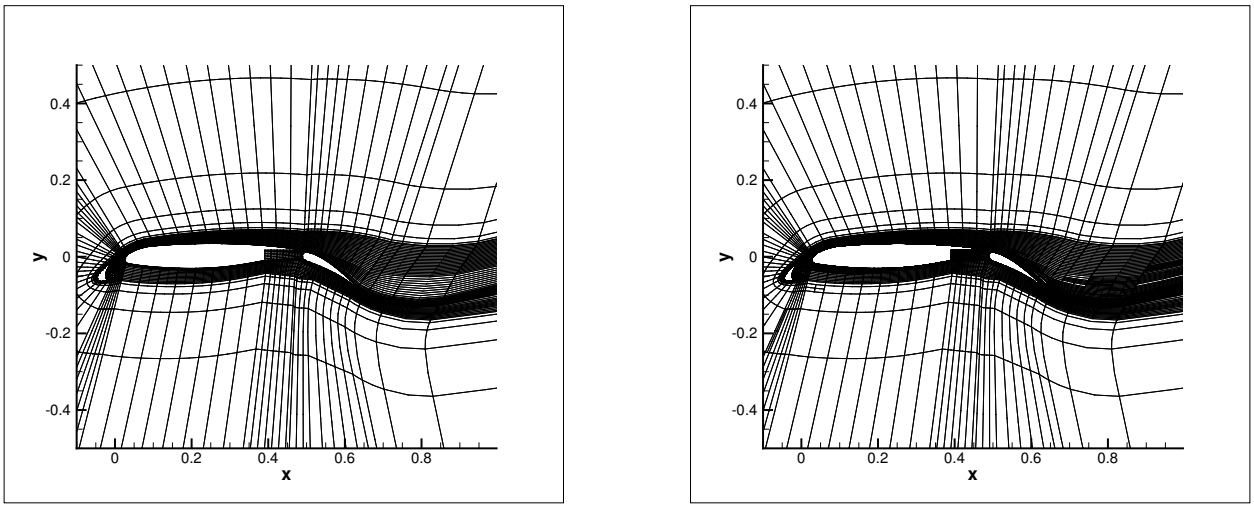


(a) $\alpha = 0^\circ$ (b) $\alpha = 10^\circ$

Figure 4.9: Skin friction coefficient C_f for turbulent flow over the NACA0012 airfoil at $M_0 = 0.15, Re_c = 6 \times 10^6$

4.3.3 Turbulent flow over the high-lift multi-element configuration 30P30N

The final test case is the turbulent flow over a high-lift multi-element configuration at the Mach number $M_0 = 0.2$, Reynolds number $Re_c = 9 \times 10^6$ based on the reference chord length of 0.5588, and an angle of attack $\alpha = 16^\circ$. The geometry is configured for a landing configuration consisting of a leading edge slat, a main airfoil and a trailing edge flap. This case is one of the benchmark problems adopted by the 1^{st} International Workshop on High-Order CFD Methods[1]. The initial mesh provided by Ceze consists of 4070 high-order quadrilateral elements as shown in Fig. 4.10(a).

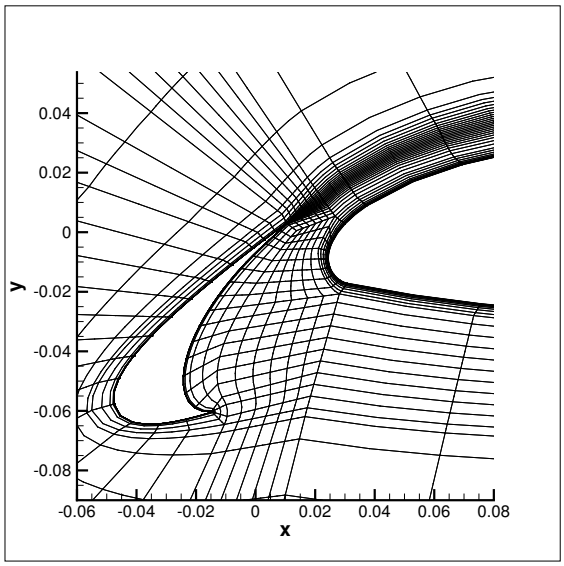


(a) The initial mesh

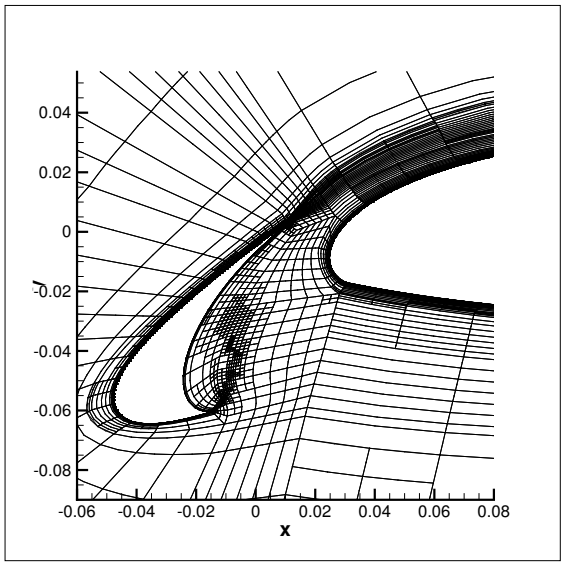
(b) The adaptive refined mesh

Figure 4.10: Mesh comparison of the adjoint-based h-adaptation for the 30P30N multi-element airfoil at $M_0 = 0.2, Re_c = 9 \times 10^6, \alpha = 16.2^\circ, p = 2$

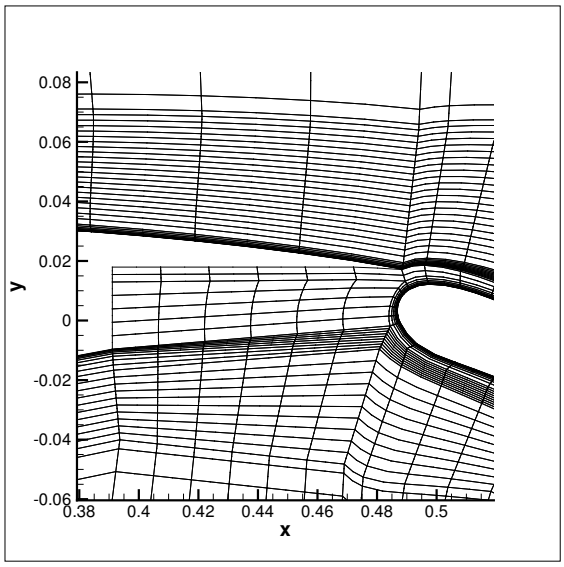
We carried out adaptive computational simulations using the $p = 1,2,3$ CPR discretizations. The drag adjoint error indicator was used to drive the isotropic h-adaptions for 5, 5, and 3 adaptations respectively, and the final numbers of cells are 17,699, 16,417, and 9,026 respectively. Fig. 4.10 - Fig. 4.11, Fig. 4.12 and Fig. 4.13 compare the meshes, the Mach number and non-dimensional eddy viscosity on the initial mesh and the final adaptive mesh using the $p = 2$ (third-order) discretization. Note that the slat wake continues all the way over the main airfoil and the flap. In addition, the maximum eddy viscosity is observed in the flap wake region, while relatively large eddy viscosity is also located in the regions of the slat wake as well as the recirculation regions near the slat and flap. To resolve the flow structures, regions inside the boundary layer, along



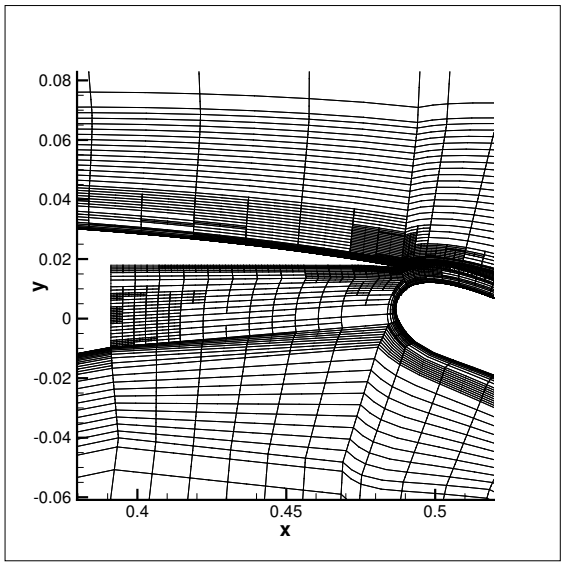
(a) The initial mesh around slat



(b) The adaptive refined mesh around slat

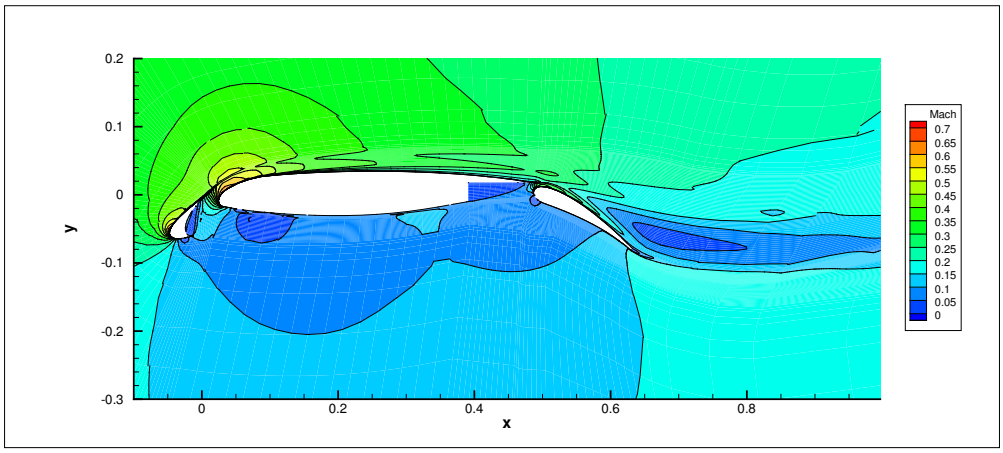


(c) The initial mesh around flap

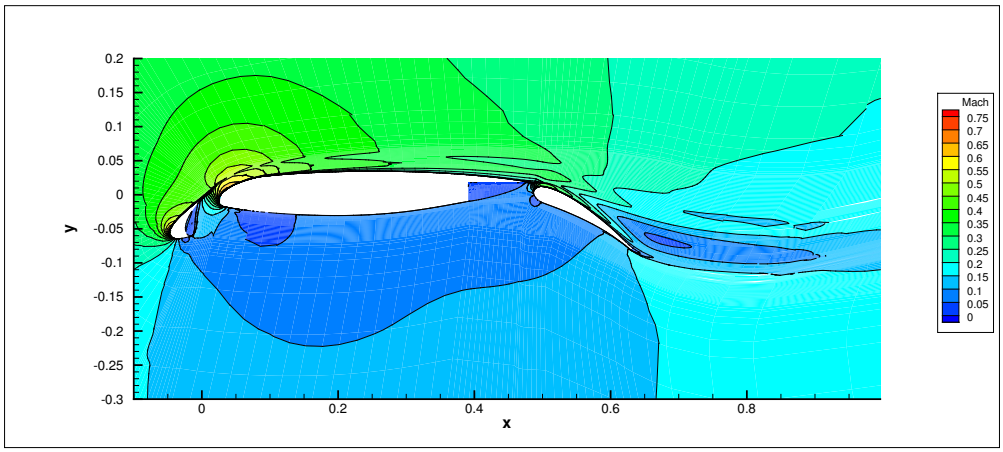


(d) The adaptive refined mesh around flap

Figure 4.11: Zoom in of mesh comparison of the adjoint-based h-adaptation for the 30P30N multi-element airfoil at $M_0 = 0.2, Re_c = 9 \times 10^6, \alpha = 16.2^\circ, p = 2$

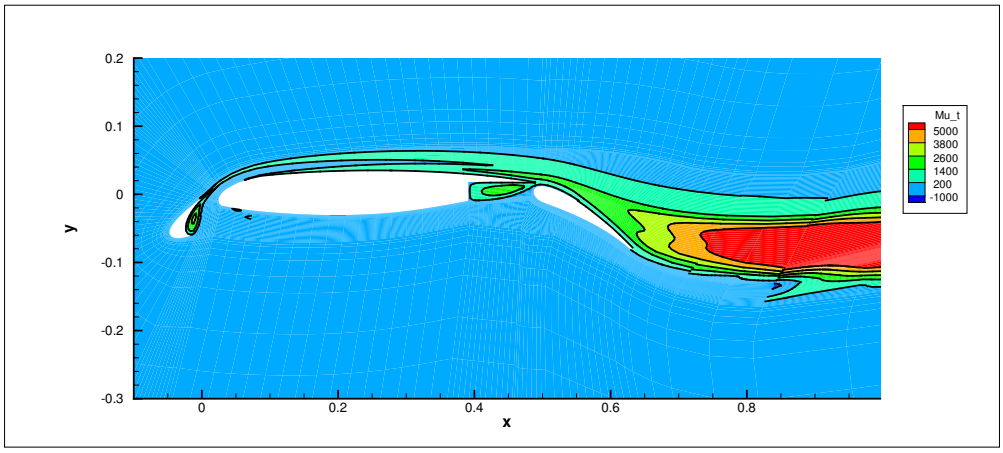


(a) Mach number contours on the initial mesh

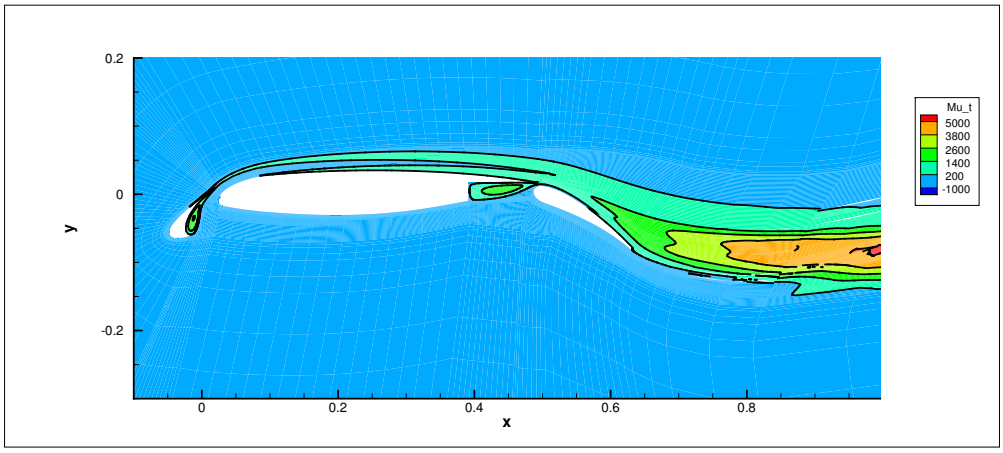


(b) Mach number contours on the adaptive refined mesh

Figure 4.12: Comparison of the Mach number contour by the adjoint-based h-adaptation for the 30P30N multi-element airfoil at $M_0 = 0.2, Re_c = 9 \times 10^6, \alpha = 16.2^\circ, p = 2$

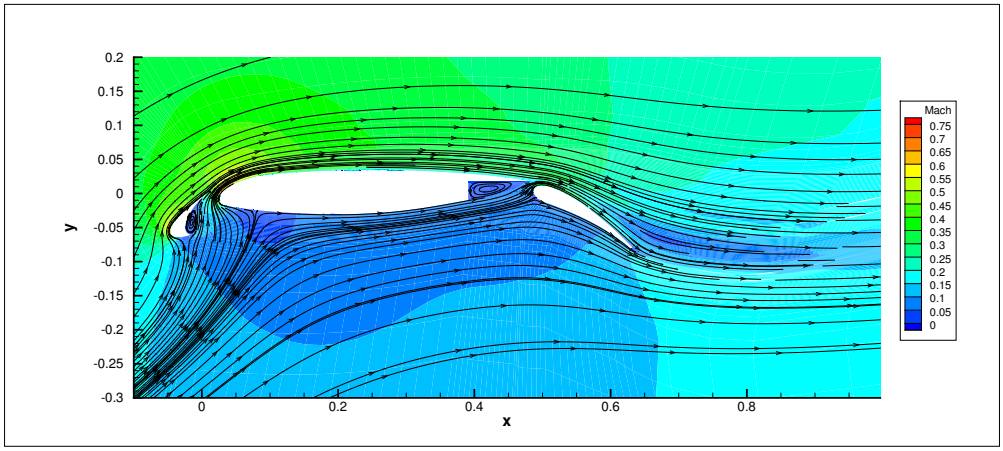


(a) Nondimensional eddy viscosity contours on the initial mesh

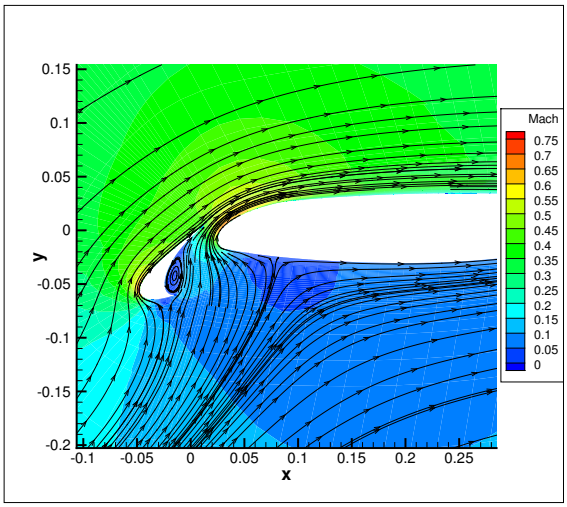


(b) Nondimensional eddy viscosity contours on the adaptive refined mesh

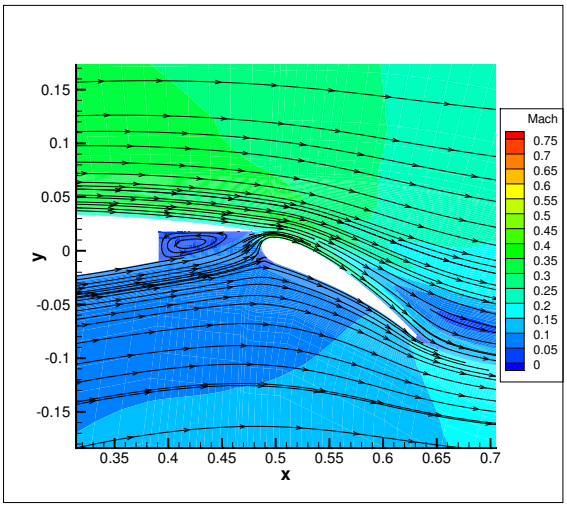
Figure 4.13: Comparison of the non-dimensional eddy viscosity by the adjoint-based h-adaptation for the 30P30N multi-element airfoil at $M_0 = 0.2, Re_c = 9 \times 10^6, \alpha = 16.2^\circ, p = 2$



(a) Streamlines around 30P30N



(b) Slat streamlines



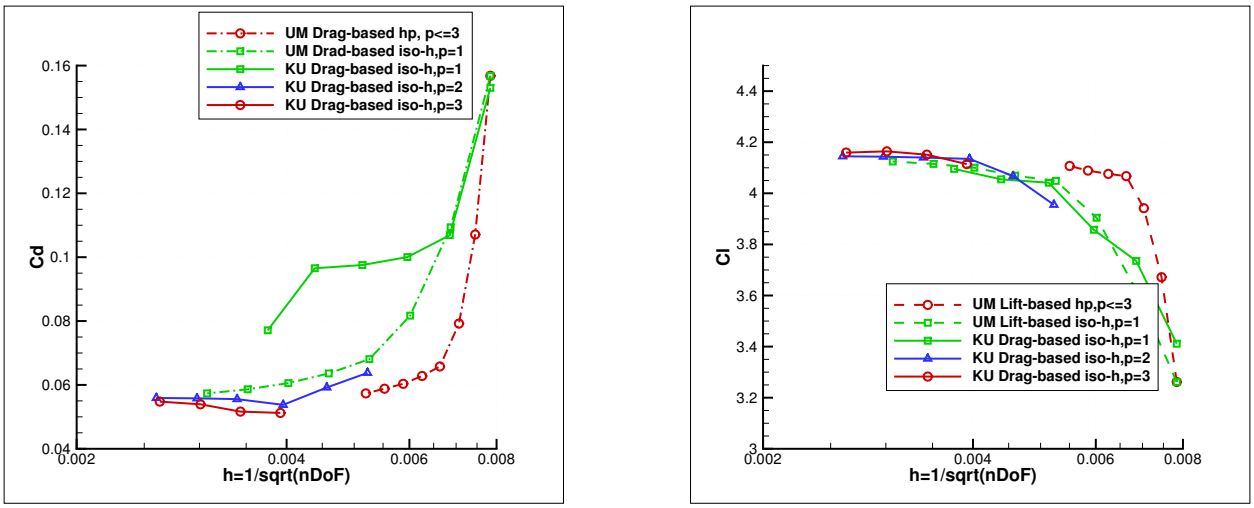
(c) Flap streamlines

Figure 4.14: Streamlines for turbulent flow over the 30P30N multi-element airfoil at $M_0 = 0.2, Re_c = 9 \times 10^6, \alpha = 16.2^\circ, p = 2$

the slat wake, the flap wake and the flap cove were targeted for mesh refinements by the adjoint-based error estimation. Also, the trailing edges of the three elements are refined repeatedly to alleviate the effect of the singularities.The necessity of h-adaptation is illustrated in the flap wake region, where the non-dimensional eddy viscosity solved on the initial mesh exhibits negative values, while no negative values are observed on the final mesh. This implies that the adjoint based h-adaptation approach is capable of capturing the under-resolved regions and putting more degrees of freedom (DOFs) there so that the functional error is efficiently reduced in terms of the number of DOFs. Fig. 4.14 displays the streamlines around the multi-element configuration. As shown in Fig. 4.14(b)(c), recirculation regions near the slat lower surface and the flap cove are resolved smoothly.

Fig. 4.15 compares the convergence of the drag and lift coefficients with other numerical results from the 2^{nd} High Order Workshop (HOW)[1]. The numerical simulations performed by the University of Michigan (UM) employed the SA turbulence model, Sutherland's law for viscosity and the same quadrilateral initial mesh as the present work (KU). As shown in Fig. 4.15(a), the drag coefficients computed by UM hp-adaptation and KU h-adaptation with $p = 1, 2, 3$ CPR discretizations converge to the same value, which demonstrates that mesh and order independent results are obtained. In terms of the lift coefficients depicted in Fig. 4.15(b), results from UM and KU are different. The $p = 1$ h-adaptation from UM and KU converged to the same value, while a slight difference exists between the results of UM hp-adaptation and KU high-order h-adaptation, which requires an further investigation.

Fig. 4.16 displays the surface pressure coefficients solved with $p = 1, 2, 3$ discretizations, along with the experimental data provided in the work by Wang[74]. In Fig. 4.16(a), there are some over-predictions and oscillations on the slat and flap respectively, using the $p = 1$ discretization, but they are not present in the solutions using the $p = 2$ and $p = 3$ discretizations, which implies that high-order discretizations result in improved solution accuracy. The surface pressure coefficient solved with the $p = 3$ discretization is further compared with other numerical results as well as experimental data in Fig. 4.16(b). The CPR discretization shows a very good agreement with the



(a) the drag coefficient C_d

(b) the lift coefficient C_l

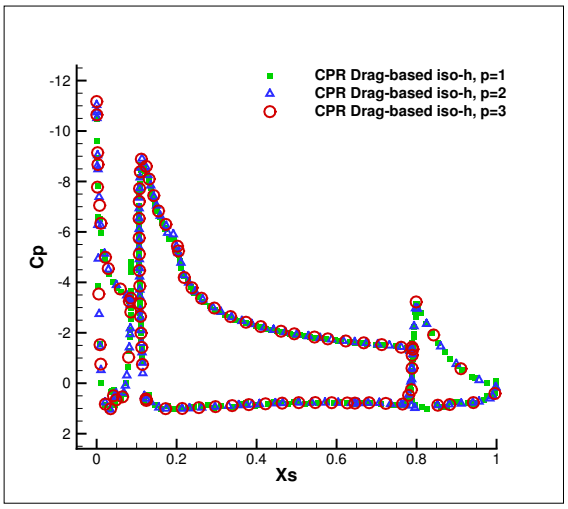
Figure 4.15: Turbulent flow over the 30P30N multi-element airfoil at $M_0 = 0.2, Re_c = 9 \times 10^6, \alpha = 16^\circ$

experimental data and the numerical result by UM.

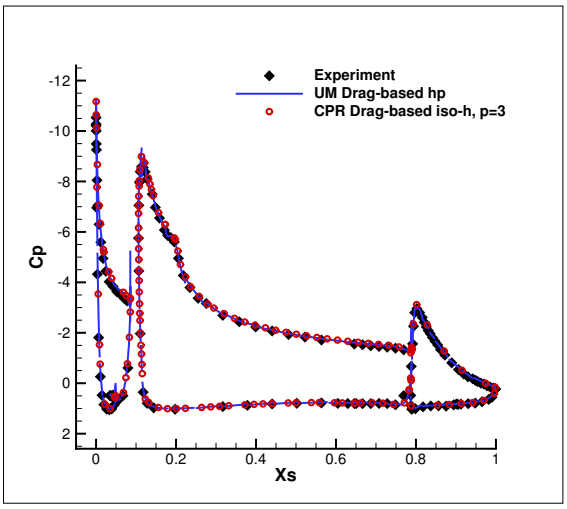
The skin friction coefficients solved with $p = 1,2,3$ discretizations are compared with each other and other numerical results in Fig. 4.17. In Fig. 4.17(a), some oscillations are present in the solutions, especially on the slat and flap using the $p = 1$ discretization. However, the solutions are enhanced using the $p = 2$ and $p = 3$ discretizations as expected. Because the disparity in the sign definition of the skin friction coefficient, the absolute value of C_f solved with the $p = 3$ discretization is further compared with other numerical results in Fig. 4.17(b). The absolute values of skin friction coefficients are in a reasonable agreement, except on the slat. The UM and KU results agree well with each other, except on the trailing points of the three elements. These numerical results of the skin friction coefficients should be compared with experimental data, if available.

4.4 Summary of Chapter

In this Chapter, the adjoint-based error estimation and mesh adaptation method is applied to the CPR discretization of the RANS equations with the modified SA model to solve the problems left

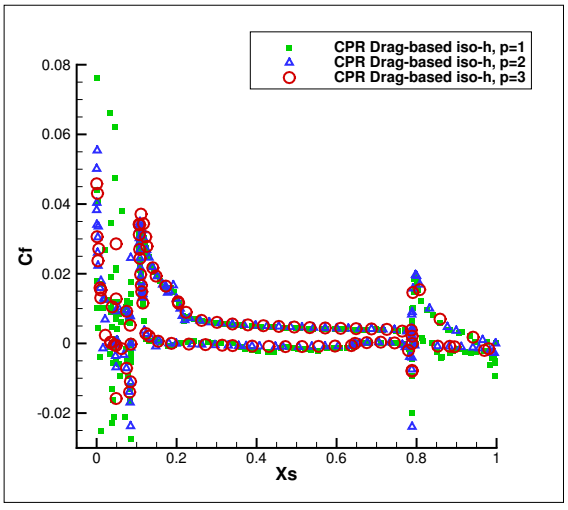


(a) CPR results with $p = 1, 2, 3$

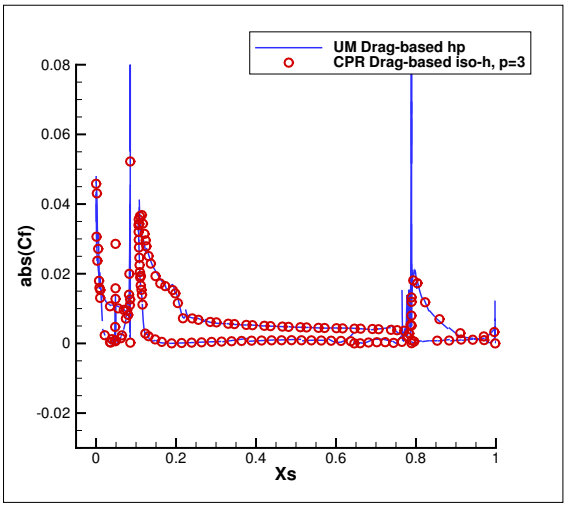


(b) Comparison of C_p

Figure 4.16: Surface pressure coefficient C_p for turbulent flow over the the 30P30N multi-element airfoil at $M_0 = 0.2, Re_c = 9 \times 10^6, \alpha = 16.2^\circ$



(a) CPR results with $p = 1, 2, 3$



(b) Comparison of C_f

Figure 4.17: Skin friction coefficient C_f for turbulent flow over the the 30P30N multi-element airfoil at $M_0 = 0.2, Re_c = 9 \times 10^6, \alpha = 16.2^\circ$

in Chapter 2: the engineering output(i.e., lift and drag coefficients) didn't achieve mesh and order independent results even with a relatively fine mesh for the turbulent flow over the NACA0012 airfoil.

Three benchmark test cases are used to verify the accuracy and efficiency of the present framework: turbulent flows over a flat plate, the NACA0012 airfoil and the three-element high-lift 30P30N configuration. The computed high-order solutions are compared to the experimental data, and other numerical simulations. Both of the h-adaptation solutions of the turbulent flow over the flat plate and the NACA0012 airfoil show an excellent agreement with the other benchmark simulations and experimental data. The adjoint-based adaptive mesh refinement method is necessary to obtain the mesh and order independent results of the drag and lift coefficients for the turbulent flow over the NACA0012 airfoil. Furthermore, the CPR RANS-SA solver is examined in capturing various flow structures for the complex 30P30N geometry accurately and efficiently. The h-adaptation results of the surface pressure and skin friction coefficients show a good agreement with experimental data and other benchmark simulations, which demonstrates the ability of the present method to efficiently reduce the functional errors in terms of the number of degrees of freedom (DOFs).

A further investigation is necessary to obtain mesh and order independent results for the drag and lift coefficients for the turbulence flow over the multi-element high-lift 30P30N configuration.

Chapter 5

Conclusions and Future Work

5.1 Summary

This thesis presents an adaptive Correction Procedure via Reconstruction (CPR) discretization of the Reynolds-Averaged Navier-Stokes (RANS) equations with the modified Spalart-Allmaras (SA) model and its application to the turbulent flow simulations.

To improve the time integration strategy, a comparative study of several well-known or recently developed implicit time integration schemes with the CPR discretization is carried out. The BLU-SGS approach, the matrix-free GMRES with BLU-SGS as a preconditioner, and the GMRES solver available in PETSc with three different preconditioners (JACOBI, ILU, LINE) are adopted in the development. The robustness and efficiency of these time integration schemes are evaluated by the simulation of inviscid and viscous steady flows. Furthermore, a line-search method is applied with the objective of improving the global convergence properties of the time integration strategy. The time integration schemes is parallelized for distributed memory system.

In space discretization realm, the high-order CPR discretization was first developed to solve the Eikonal equation to determine the distance of each solution point to the nearest curved polynomial wall boundaries, which is related with the non-dimensional length scale in the SA model. Robust and efficient simulations are observed when the CPR discretization is applied to challenging

problems with complex geometries.

A dual-consistent high-order CPR discretization is then developed to solve the RANS equations with the modified SA model. An adjoint-based adaptive mesh refinement method is utilized to minimize the output error. The accuracy of the present framework is demonstrated for several benchmark turbulent flow problems by comparing with other simulations and experimental data. In addition, the comparison with the numerical results of the uniform mesh refinement method demonstrated the present mesh adaptation method to be capable of efficiently reduce the functional errors in terms of the number of degrees of freedom.

5.2 Conclusions

The test cases in Chapter 3 show that, for inviscid and viscous flow, GMRES with BLU-SGS as the preconditioner is demonstrated the most efficient and economical. However, in sense of robustness, GMRES solver solving the fully linearized system with ILU(1) preconditioner is recommended for viscous flow.

The present CPR dicretization of the Eikonal equation is demonstrated to be accurate and efficient by the four test cases in Section 2.3.3: the square case surrounded by wall boundaries demonstrates the capability of the CPR Eikonal solver on internal flow geometries; The super convergence given by the ring case shows the accuracy and efficiency of the proposed high-order solver; The smooth high-order wall distance solutions of the RAE2282 airfoil and the high-lift 30P30N multi-element configuration further demonstrate the robustness of the proposed solver.

The uniform mesh refinement results of two benchmark test cases in Chapter 2 show the accuracy of the high-order CPR discretization of the RANS equations. The computed high-order solutions are compared to the experimental data, the CFL3D and FUN3D results. The converged drag coefficient of the flat plate case agrees with CFL3D and FUN3D results within 0.1 count, and the Richardson extrapolation of the drag co- efficient numerically verify the order of accuracy of the present high-order CPR discretization of the RANS equations. In terms of the skin friction

coefficient on the flat plate and the NACA0012 airfoil, the mesh and order independent results are obtained and show an excellent agreement with the other benchmark simulations. However, the mesh and order independence of the drag and lift coefficients is not clear for the NACA0012 airfoil test case, especially at high angles of attack, which leads to the discussion of adjoint-based error estimation and mesh adaptation method in Chapter 4.

The adaptive CPR discretization of the RANS equations is capable of efficiently reduce the functional errors in terms of the number of degrees of freedom comparing to the uniform mesh refinement. This is demonstrated by the numerical results of three benchmark test cases in Chapter 4: turbulent flows over a flat plate, the NACA0012 airfoil and the three-element high-lift 30P30N configuration. Both of the h-adaptation solutions of the turbulent flow over the flat plate and the NACA0012 airfoil show an excellent agreement with the other benchmark simulations and experimental data. The adjoint-based adaptive mesh refinement method is necessary to obtain the mesh and order independent results of the drag and lift coefficients for the turbulent flow over the NACA0012 airfoil. Furthermore, the CPR discretization of the RANS equations with the SA model is examined in capturing various flow structures for the complex 30P30N geometry accurately and efficiently. The h-adaptation results of the surface pressure and skin friction coefficients show a good agreement with experimental data and other benchmark simulations.

5.3 Suggestions for Future Research

Certain topics for future work were identified during the course of this work. These topics are listed below:

- Improve the robustness of time integration schemes by physicality constrained solver.
- Investigate other turbulence models for high-order CPR discretization to improve the robustness of CPR method for more complicated turbulent flows.
- Apply the adaptive high-order CPR discretization of RANS equations to 3D challenging

problems in the aeronautical industry.

- Develop detached eddy simulation (DES) for CPR method by combining the large eddy simulation (LES) with the present RANS discretization.

- Research in *h* and *p* coarsening that allow for a reallocation of computation cost in the mesh, instead of increments of computation cost at each adaptive step.

References

[1] (2012). 1^{st} international workshop on high-order CFD methods. <*http://zjwang.com/hiocfd.html*>.

[2] (Lagnley Research Center). Turbulence modeling resource. <*http://turbmodels.larc.nasa.gov/*>.

[3] Ainsworth, M. & Oden, J. (1993). A unified approach to a posteriori error estimation using element residual methods. *Numerische Mathematik*, 65(1), 23–50.

[4] Allmaras, S. R. (1999). Multigrid for the 2-d compressible Navier-Stokes equations. *AIAA paper 1999-3336*.

[5] Baker, T. J. (1997). Mesh adaptation strategies for problems in fluid dynamics. *Finite Elements in Analysis and Design*, 25(3-4), 243–273.

[6] Balay, S., Buschelman, K., Gropp, W., Kaushik, D., Knepley, M., McInnes, L., Smith, B., & Zhang, H. (2001). Petsc webpage. <*http://www.mcs.anl.gov/petsc*>.

[7] Balay, S., Gropp, W. D., McInnes, L. C., & Smith, B. F. (1997). Efficient management of parallelism in object-oriented numerical software libraries. In *Modern software tools for scientific computing* (pp. 163–202). Springer.

[8] Barth, T. (1998). Numerical methods for gasdynamic systems on unstructured meshes. *An Introduction to Recent Developments in Theory and Numerics for Conservation Laws*, 5, 195–285.

[9] Bassi, F. & Rebay, S. (1997a). A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 131(2), 267–279.

[10] Bassi, F. & Rebay, S. (1997b). High-order accurate discontinuous finite element solution of the 2D Euler equations. *Journal of Computational Physics*, 138(2), 251–285.

[11] Bassi, F. & Rebay, S. (2000). Gmres discontinuous Galerkin solution of the compressible Navier-Stokes equations. *Lecture Notes in Computational Science and Engineering*, 11, 197–208.

[12] Becker, R. & Rannacher, R. (1996). A feed-back approach to error control in finite element methods: Basic analysis and examples. *East-West J. Numer. Math*, 4, 237–264.

[13] Becker, R. & Rannacher, R. (2001). An optimal control approach to a posteriori error estimation in finite element methods. *Acta Numerica*, 10, 1–102.

[14] Berger, M. & Colella, P. (1989). Local adaptive mesh refinement for shock hydrodynamics. *Journal of computational Physics*, 82(1), 64–84.

[15] Brown, D. A., Buckley, H. P., Osusky, M., & Zingg, D. W. (2015). Performance of a newton–krylov–schur algorithm for the numerical solution of the steady Reynolds-averaged Navier-Stokes equations. *AIAA Paper 2015-1744*.

[16] Burgess, N. K. & Mavriplis, D. J. (2012). Robust computation of turbulent flows using a discontinuous Galerkin method. *AIAA paper*, 457.

[17] Burgess, N. K., Nastase, C. R., Mavriplis, D. J., & Martinelli, L. (2010). Efficient solution techniques for discontinuous Galerkin discretizations of the Navier-Stokes equations on hybrid anisotropic meshes. *AIAA paper*, 1448.

[18] Cagnone, J. & Nadarajah, S. (2012). A stable interface element scheme for the p-adaptive lifting collocation penalty formulation. *Journal of Computational Physics*, 231(4), 1615 – 1634.

[19] Celik, I. B., Ghia, U., Roache, P. J., et al. (2008). Procedure for estimation and reporting of uncertainty due to discretization in CFD applications. *Journal of fluids Engineering-Transactions of the ASME*, 130(7).

[20] Ceze, M. & Fidkowski, K. J. (2012). Anisotropic hp-adaptation framework for functional prediction. *AIAA Journal*, 51(2), 492–509.

[21] Ceze, M. A. d. B. (2013). A robust hp-adaptation method for discontinuous Galerkin discretizations applied to aerodynamic flows. *PhD thesis*.

[22] Chen, R. & Wang, Z. (2000). Fast, block lower-upper symmetric Gauss-Seidel scheme for arbitrary grids. *AIAA journal*, 38(12), 2238–2245.

[23] Cockburn, B., Lin, S., & Shu, C. (1989). TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One-dimensional systems. *Journal of Computational Physics*, 84(1), 90–113.

[24] Crivellini, A., D'Alessandro, V., & Bassi, F. (2013). A Spalart-Allmaras turbulence model implementation in a discontinuous Galerkin solver for incompressible flows. *Journal of Computational Physics*, 241, 388–415.

[25] Diosady, L. T. & Darmofal, D. L. (2009). Preconditioning methods for discontinuous Galerkin solutions of the Navier-Stokes equations. *Journal of Computational Physics*, 228(11), 3917–3935.

[26] Diskin, B., Thomas, J., Rumsey, C., & Schwöppe, A. (2015). Grid convergence for turbulent flows. *AIAA Paper 2015-1746*.

[27] Fares, E. & Schröder, W. (2002). A differential equation for approximate wall distance. *International journal for numerical methods in fluids*, 39(8), 743–762.

[28] Fidkowski, K. (2011). Review of output-based error estimation and mesh adaptation in computational fluid dynamics. *AIAA Journal*, 49(4), 673–694.

[29] Fidkowski, K. & Darmofal, D. (2007). A triangular cut-cell adaptive method for high-order discretizations of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 225(2), 1653–1672.

[30] Fidkowski, K. & Roe, P. (2009). Entropy-based mesh refinement, I: The entropy adjoint approach.

[31] Fidkowski, K. & Roe, P. (2010). An entropy adjoint approach to mesh refinement. *SIAM Journal on Scientific Computing*, 32(3), 1261–1287.

[32] Gao, H. & Wang, Z. J. (2011). A residual-based procedure for hp-adaptation on 2D hybrid meshes. *AIAA Paper 2011-492*.

[33] Giles, M. & Pierce, N. (1997). Adjoint equations in CFD: duality, boundary conditions and solution behaviour. *AIAA paper 97-1850*.

[34] Giles, M. & Pierce, N. (2003). Adjoint error correction for integral outputs. In T. Barth & H. Deconinck (Eds.), *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, volume 25 of *Lecture Notes in Computational Science and Engineering* (pp. 47–95). Springer Berlin Heidelberg.

[35] Godunov, S. K. (1959). A finite-difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics. *Mat. Sb.*, 47, 271.

[36] Haga, T., Sawada, K., & Wang, Z. (2009). An implicit LU-SGS scheme for the spectral volume method on unstructured tetrahedral grids. *Communications in Computational Physics*, 6(5), 978.

[37] Harris, R. & Wang, Z. J. (2009). High-order adaptive quadrature-free spectral volume method on unstructured grids. *Computers & Fluids*, 38(10), 2006–2025.

[38] Hartmann, R. & Houston, P. (2002). Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *Journal of Computational Physics*, 183(2), 508 – 532.

[39] Hovland, P. D., McInnes, L. C., et al. (2001). Parallel simulation of compressible flow using automatic differentiation and petsc. *Parallel Computing*, 27(4), 503–519.

[40] Huang, W. & Russell, R. D. (2010). *Adaptive Moving Mesh Methods*, volume 174. Springer.

[41] Huynh, H. T. (2007). A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods. *AIAA Paper 2007-4079*.

[42] Huynh, H. T. (2009). A reconstruction approach to high-order schemes including discontinuous Galerkin for diffusion. *AIAA Paper 2009-403*.

[43] Huynh, H. T., Z. J. W. & Vincent, P. E. (2014). High-order methods for computational fluid dynamics: A brief review of compact differential formulations on unstructured grids. *Computers & Fluids*, 98(0), 209–220.

[44] Jameson A, C. D. (2001). How many steps are required to solve the Euler equations of steady compressible flow: in search of a fast solution algorithm. *15th AIAA computational fluid dynamics conference*.

[45] Johnson, C. (1998). Adaptive finite element methods for conservation laws. *Advanced numerical approximation of nonlinear hyperbolic equations*, (pp. 269–323).

[46] Kopriva, D. A. (1996). A conservative staggered-grid chebyshev multidomain method for compressible flows. II. a semi-structured method. *Journal of Computational Physics*, 128(2), 475 – 488.

[47] Krist, S. L., Biedron, R. T., & Rumsey, C. L. (1998). Cfl3d user's manual (version 5.0). *NASA Technical Report 1988-208444*.

[48] Leer, B. V. (1979). Towards the ultimate conservative difference scheme V. a second order sequel to godunov's method. *Journal of Computational Physics*, 32, 101–136.

[49] Li, Y., Allaneau, Y., & Jameson, A. (2010). Continuous adjoint approach for adaptive mesh refinement. *AIAA Paper 2010-3982*.

[50] Liou, M.-S. (2006). A sequel to ausm, part II: Ausm+ for all speeds. *Journal of Computational Physics*, 214, 137–170.

[51] Liu, C.-B., Nithiarasu, P., & Tucker, P. (2010). Wall distance calculation using the Eikonal/Hamilton-Jacobi equations on unstructured meshes: A finite element approach. *Engineering Computations*, 27(5), 645–657.

[52] Mavriplis, D. J. (1990). Adaptive mesh generation for viscous flows using triangulation. *Journal of computational Physics*, 90(2), 271–291.

[53] Mayeur, J., Dumont, A., Destarac, D., & Gleize, V. (2015). RANS simulations on TMR test cases and M6 wing with the Onera elsa flow solver. *AIAA Paper 2015-1745*.

[54] Moro, D., Nguyen, N., & Peraire, J. (2011). Navier-Stokes solution using hybridizable discontinuous Galerkin methods. *AIAA paper*, 3407, 2011.

[55] Nguyen, N. C., Persson, P.-O., & Peraire, J. (2007). RANS solutions using high order discontinuous Galerkin methods. *AIAA Paper*, 914, 2007.

[56] Oliver, T. A. (2008). *A high-order, adaptive, discontinuous GAlerkin finite element method for the REynolds-averaged NAvier-STokes equations*. Technical report, Massachusetts Institute of Technology.

[57] Oliver, T. A. & Darmofal, D. L. (2007). An unsteady adaptation algorithm for discontinuous Galerkin discretizations of the RANS equations. *AIAA Paper*, 3940, 2007.

[58] Park, M. (2002). Adjoint-based three-dimensional error prediction and grid adaptation. *AIAA paper 2002-3286*.

[59] Pennestrì, E. & Stefanelli, R. (2007). Linear algebra and numerical algorithms using dual numbers. *Multibody System Dynamics*, 18(3), 323–344.

[60] Rasetarinera, P. & Hussaini, M. (2001). An efficient implicit discontinuous spectral Galerkin method. *Journal of Computational Physics*, 172(2), 718–738.

[61] Roe, P. L. (1981). Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43, 357–372.

[62] Rusanov, V. V. (1961). Calculation of interaction of non-steady shock waves with obstacles. *Journal of Computational Physics*, 1, 261–279.

[63] Saad, Y. & Schultz, M. H. (1986). Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, 7(3), 856–869.

[64] Schoenawa, S. & Hartmann, R. (2014). Discontinuous Galerkin discretization of the Reynolds-averaged Navier-Stokes equations with the shear-stress transport model. *Journal of Computational Physics*, 262, 194–216.

[65] Shi, L. & Wang, Z. J. (2014). Adjoint-based error estimation and mesh adaptation for the correction procedure via reconstruction method. *Submitted to Journal of Computational Physics*.

[66] Shi, L., Zhou, C., & Wang, Z. J. (2016). Adaptive RANS solution with the high-order correction procedure via reconstruction method. *AIAA Paper 2016-1826*.

[67] Shih, T. & Qin, Y. (2007). A posteriori method for estimating and correcting grid-induced errors in CFD solutions-part 1: Theory and method. *AIAA Paper 2007-100*.

[68] Spalart, P. R. & Allmaras, S. R. (1992). A one-equation turbulence model for aerodynamic flows. *La Recherche Aerospatiale*, (pp. 5–21).

[69] Steger, J. L. & Warming, R. (1981). Flux vector splitting of the inviscid gasdynamic equations with application to finite-difference methods. *Journal of computational physics*, 40(2), 263–293.

[70] Sun, Y., Wang, Z., Liu, Y., & Chen, C. (2007). Efficient implicit LU-SGS algorithm for high-order spectral difference method on unstructured hexahedral grids. *AIAA paper*, 313, 2007.

[71] Venditti, D. & Darmofal, D. (2003). Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows. *Journal of Computational Physics*, 187(1), 22–46.

[72] Venditti, D. A. & Darmofal, D. L. (2000). Adjoint error estimation and grid adaptation for functional outputs: application to quasi-one-dimensional flow. *Journal of Computational Physics*, 164(1), 204–227.

[73] Visbal, M. R. & Gaitonde, D. V. (2002). On the use of higher-order finite-difference schemes on curvilinear and deforming meshes. *Journal of Computational Physics*, 181(1), 155–185.

[74] Wang, L., Anderson, W. K., Erwin, J. T., & Kapadia, S. (2012). Solutions of high-order methods for three-dimensional compressible viscous flows. *AIAA paper 2012-2836*.

[75] Wang, L. & Mavriplis, D. (2009a). Adjoint-based hp-adaptive discontinuous Galerkin methods for the 2D compressible Euler equations. *Journal of Computational Physics*, 228(20), 7643–7661.

[76] Wang, L. & Mavriplis, D. J. (2009b). Adjoint-based hp-adaptive discontinuous Galerkin methods for the 2D compressible Euler equations. *Journal of Computational Physics*, 228(20), 7643–7661.

[77] Wang, Z. J. (2011). *Adaptive High-Order Methods in Computational Fluid Dynamics*. World Scientific Publishing.

[78] Wang, Z. J., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H. T., Kroll, N., May, G., Persson, P.-O., van Leer, B., & Visbal, M. (2013). High-order CFD Methods: Current Status and Perspective. *International Journal for Numerical Methods in Fluids*, 72, 811–845.

[79] Wang, Z. J. & Gao, H. (2009). A unifying lifting collocation penalty formulation including

the discontinuous Galerkin, spectral volume/difference methods for conservation laws on mixed grids. *Journal of Computational Physics*, 228, 8161–8186.

[80] Wang, Z. J. & Liu, Y. (2002). Spectral (finite) volume method for conservation laws on unstructured grids II. extension to two-dimensional scalar equation. *Journal of Computational Physics*, 179(2), 665–697.

[81] Warren, G., Anderson, W., Thomas, J., & Krist, S. (1991). Grid convergence for adaptive methods. In *10th AIAA Computational Fluid Dynamics Conference*.

[82] Xia, H. & Tucker, P. G. (2010). Finite volume distance field and its application to medial axis transforms. *International journal for numerical methods in engineering*, 82(1), 114–134.

[83] Yano, M. & Darmofal, D. L. (2012). An optimization-based framework for anisotropic simplex mesh adaptation. *Journal of Computational Physics*, 231(22), 7626–7649.

[84] Yoon, S. & Jameson, A. (1988). Lower-upper symmetric-Gauss-Seidel method for the Euler and Navier-Stokes equations. *AIAA journal*, 26(9), 1025–1026.

[85] Zhang, X., Vallet, M.-G., Dompierre, J., Labbe, P., Pelletier, D., & Trepanier, J.-Y. (2001). Mesh adaptation using different error indicators for the Euler equations. *AIAA Paper 2001-2549*.

[86] Zhou, C. & Wang, Z. (2013). An evaluation of implicit time integration schemes for discontinuous high order methods. *AIAA Paper 2013-2688*.